WatchMI: Applications of Watch Movement Input on Unmodified Smartwatches

Hui-Shyong Yeo
School of Computer Science,
University of St Andrews
Fife, Scotland, UK
hsy@st-andrews.ac.uk

Andrea Bianchi
Department of Industrial Design,
KAIST
Daejeon, Republic of Korea
andrea@kaist.ac.kr

Juyoung Lee
Graduate School of Culture
Technology, KAIST
Daejeon, Republic of Korea
ejuyoung@kaist.ac.kr

Aaron Quigley
School of Computer Science,
University of St Andrews
Fife, Scotland, UK
aquigley@st-andrews.ac.uk

MobileHCI '16 Adjunct, September 06-09, 2016, Florence, Italy.
ACM 978-1-4503-4413-5/16/09.
http://dx.doi.org/10.1145/2957265.2961825

Abstract

In this demo, we show that it is possible to enhance touch interaction on unmodified smartwatch to support continuous pressure touch, twist and pan gestures, by only analyzing the real-time data of Inertial Measurement Unit (IMU). Our evaluation results show that the three proposed input interfaces are accurate, noise-resistant, easy to use and can be deployed to a variety of smartwatches. We then showcase the potential of this work with seven example applications. During the demo session, users can try the prototype.

Author Keywords

Rich Touch; Smart watch; Small screen; Wearable devices

ACM Classification Keywords

H.5.2. [Information interfaces and presentation]: User interfaces - Input devices and strategies;

Introduction

Many of the smartwatches and fitness trackers have touch screen displays for interaction, such as the Android Wear, Fitbit Surge or Apple Watch [2]. However, interacting with these devices can be difficult, mainly due to the small interaction surface and relatively large human fingers. Furthermore, existing problems on mobile devices such as the “fat finger” [7] or occlusion [9] problems are exacerbated with the small interaction surfaces on these wearable devices.

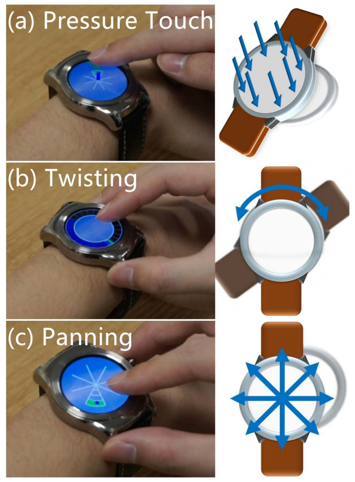

Figure 1: WatchMI interaction. From top to bottom: Pressure touch, twist and panning.

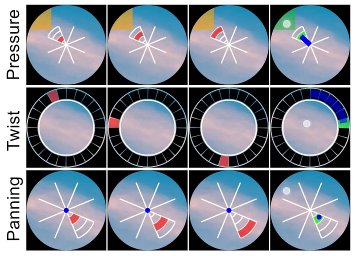

Figure 2: User study: control the blue indicator and remain in the red target for 1 second.

In this demo, we show a new approach to sense continuous rate-based “touch pressure”, “twist angle” and “pan movement” on unmodified smartwatches. With our techniques, users can interact with small devices without suffering the limitations of the screen size, by taking advantage of the expanded input expressiveness. Our approach is self-contained, as it leverages the built-in IMU already available in almost every smartwatch and smart wristband in the consumer market. Thus, it does not require any additional sensor or hardware modification of the watch and can already work on most of the wearable devices in the market, e.g., via a simple software update. The software utilizes a sensor-fusion approach combining touch events with the orientation of the watch to enrich the sensing capabilities, while also effectively rejecting false positives. In the following sections we introduce related work, a description of our system, show case a variety of example applications and illustrate limitations and future work.

Related Work

Previous attempts to address these challenges often rely on incorporating new input modalities such as, pressure sensitive touch [2, 5], bezel rotation [6] or a combination of them [4, 10]. Xiao et.al. [10] and SkinWatch [4] enhance the interaction by supporting tilt, twist and pan on the watchface with Hall-effect joysticks or photo-reflective sensors. Additional sensors can also extend the interaction space around the watch. For example, SkinButtons [3] expands the input area to the skin surface beside the watch. Such approaches, however, require additional hardware which can increase both cost and weight of the devices. The need for additional hardware or moving parts also limits their adoption and exploration for common use, and is susceptible to water damage [4, 10]. Nonetheless, it is possible to achieve higher input expressiveness that is usually not possible on off-the-shelf devices.

Prototype & Interaction

To showcase our system, we prototype and deployed the software in several popular Android Wear watches such as LG Urbane, Samsung Gear, LG G Watch and Moto 360. We use only Android SDK for Android Wear. The system adds omni-direction pressure touch, bi-direction twist and omni-direction pan that, to a user’s eyes, work like many familiar devices, such as music keyboards (pressure touch), volume knobs (twist) or joysticks (panning).

The working principle is simple. The human skin is elastic and stretchable, and so are typical watch straps made from leather, rubber or metal meshes. When users press or twist the watchface with force, the skin and watch strap are deformed. We can then measure the changes in watch’s tilt angle due to this deformation, and then use it to infer the amount of pressure and the direction of the applied force. Because of the skin and strap elasticity, when users release the force, the watchface naturally goes back to its original position. This "natural" force also gives tactile feedback.

The software collects in real-time the rotation vector [1] from the IMU sensor (a measure of the device orientation based on the fusion data from the internal accelerometer, gyroscope and magnetometer), at a sampling rate of about 100 Hz. The rotation vector gives a Quaternion value but can be easily converted to yaw, pitch and roll using the provided Android SDK. When the user first touches the screen, the software records the current orientation of the watch as the initial value. Then the software constantly measures and calculates the difference against the latest value as a delta vector, which is the changes of the watch orientation. This delta of relative change is then normalized and linearly mapped to screen coordinates, with a minimum threshold and multiplier pre-determined in a pilot test. For pressure touch and panning, we use only the pitch and roll

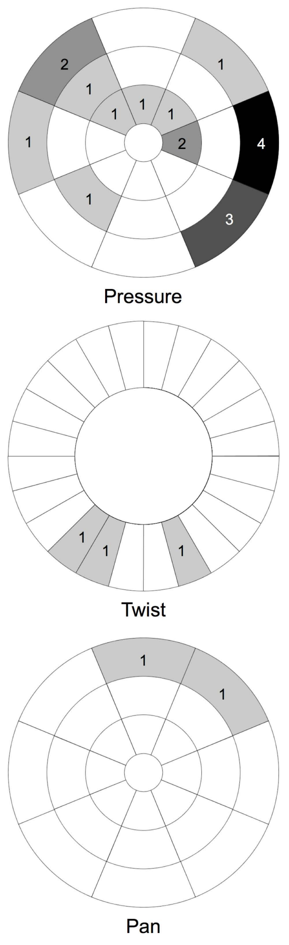

Figure 3: Occurrences of errors performed by the 12 participants.

value while in twisting we use only the yaw value. Finally, depending on the technique being used, we visualize the delta as a growing bar (pressure), rotating arc (twist) and a cursor (panning) that moves in a 2D plane.

Since the IMU senses any changes in acceleration or orientation, which can be caused by any hand motion, we use the touch event as a trigger, and stop measuring the value when the user lifts the finger from the screen. This simple yet effective approach of combining touch and orientation makes the system resistant to noise (e.g., accidental arm movements) and allows users to comfortably input on the smartwatch, regardless its initial orientation. When no touch is detected, the changes in IMU values are simply discarded, preventing us interpreting unwanted motions. In addition, it allows us to turn off IMU sensing when not needed, reducing battery consumption. Finally in the pressure touch mode, we cross check both the touch location and the tilt direction to further improve the accuracy.

Evaluation Study

We designed a user study to assess the usability of the three aforementioned input interfaces - pressure, twist and pan input. We recruited 12 participants. We modified the three interfaces to collect measurement for input entry times and errors (Figure 2). Overall, participants were able to complete all tasks in about 2.4 seconds (including 1s dwell time) and with an average error rate of 0.7%. During the interviews, most participants described the system as “responsive” and “easy to learn”, findings which were numerically corroborated in the post-experiment questionnaire. Finally, it is clear from Figure 3 that many errors occurred on i) the top left and ii) right hemisphere of the watch. It is because when clicking on the top left portion of the screen, the finger in contact with the screen occluded portions of the screen where the visual target indicator was displayed (assuming wearing the watch on left hand). The errors on the right are caused by wearing the watch too tightly, where the ulnar bone blocks and limits the pressure applied by the user. For more information about the study, we refer readers to the WatchMI paper [11].

Example Applications

To illustrate the potential and immediate feasibility of our approach, we developed 7 applications (Figure 4) to showcase these 3 interaction techniques - a) Continuous map navigation b) Alarm clock c) Music player d) Pan gesture recognition e) Text entry f) File explorer and g) Controlling remote devices. These demos take advantage of the rate control based on the strength applied, such as continuously navigating and zooming the map, twisting the clock hand, turning up volume or controlling game characters.

a) In the ***map*** application, users can twist the watchface in both directions (clockwise and anticlockwise) to zoom-in or zoom-out. Furthermore, by pressing any side of the watch, users can move through the map continuously at different speed based on the amount of pressure applied. Therefore, users can navigate a map easily without the need to perform multiple swipes or pinches to zoom.

b) In the ***alarm clock*** application, users can touch any part of the screen, and after a one second delay, it enters alarm setting page. Then, users can twist the watchface to manipulate the minute hand. Similarly, the minute hand is continuously rate-controlled. Its rotation speed is proportional to the twist amount applied. When users are done setting, just release the touch and it goes back to normal clock page.

c) In the ***music player*** application, there are 3 buttons (volume, playback speed and seeking). Touching each button and then twist will activate different functions accordingly. Twisting clockwise increases the value, and vice-versa.

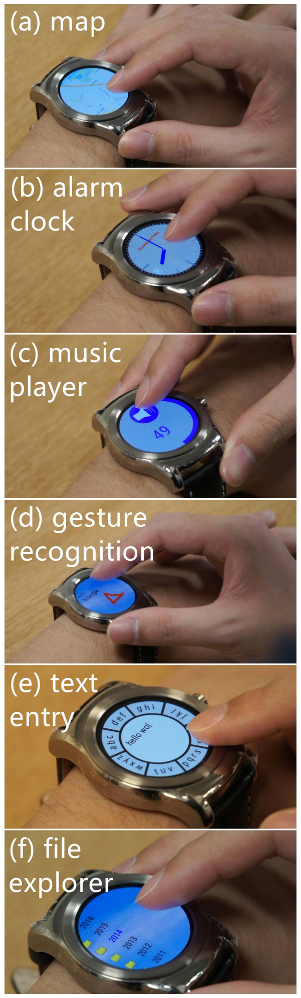

Figure 5: Demonstration of the applications. From top to bottom according to Figure 4.

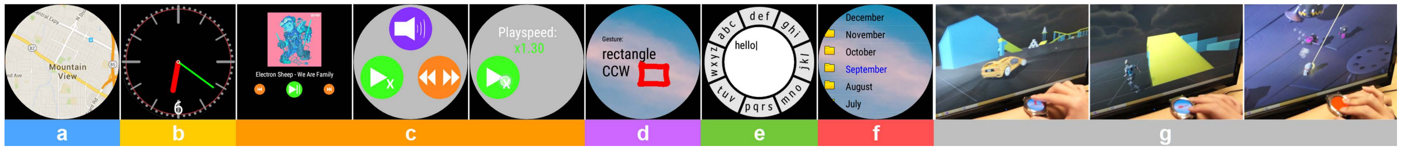

Figure 4: Screenshots of the applications. From left to right: (a) map, (b) alarm clock, (c) music player, (d) gesture recognizer, (e) text entry, (f) file explorer and (g) controlling remote devices including car, humanoid and first person shooter.

d) In the pan ***gesture recognition*** example, users can pan the watchface to draw a desired shape (rectangle, triangle, pigtail, star, etc.). The shape is then recognized by 1-dollar gesture recognizer [8]. One can imagine assigning different shapes to shortcut commands, such as drawing “f” to launch Facebook application instantly.

e) In the ***text entry*** example, the 26 English characters are strategically arranged in the perimeter (bezel) of the watch, and are grouped into 3 for each button. Users can touch any of the buttons and vary the applied pressure to disambiguate between characters in the same button. For example, in the first button, varying the pressure between light, medium and strong will insert ‘a’, ‘b’, and ‘c’, respectively.

f) In the ***file explorer*** application, users can continuously scroll through a list of folders or files using panning gesture instead of the traditional swiping action. Panning right/left will enter/exit the folder or file accordingly.

g) For ***controlling remote devices or characters***, we implemented 3 sample games (Figure 4g) in Unity game engine. i) Users can twist the watchface to steer a car’s wheel while pressing the top/bottom edge of the screen will accelerate/brake depends on the amount of pressure applied. Combining these two techniques allows users to move and steer the car at the same time. ii) Users can control a first-person humanoid character. Panning the watchface moves the character in 2D direction, the speed is depending on the pressure level. Twisting will turn the first-person camera view. iii) In a 3rd person shooting game, the control is very similar to the second game except another finger is needed to activate the gun shooting (2 fingers touch).

Our techniques, instead of strictly using for new interactions, can be used for reserved functions. For example, all Android smartwatches reserved the left and down swipe for OS functionality - exiting app and showing top menu bar. This severely limits the interaction capability. Using our techniques, perhaps assigning pressure touch on the left side of the watch for exiting app, can free the reserved left swipe for more basic purposes.

Limitations & Future Work

This work has a number of limitations but also opportunities for improvement in future revisions. First of all, our techniques require a touch event on the capacitive screen in order to discriminate intentional input from accidental arm movements. This limitation has practical consequences: capacitive input is required therefore the watch screen area is partially occluded during input interaction. Future work will attempt to address this problem by using other means of trigger, perhaps a conductive tape on the rim to route the touch, or a spike in linear acceleration and microphone input caused by user hitting the side of watch.

Our applications showcased that it is possible to combine and seamlessly integrate the three techniques, but future work will be needed to address in more detail possible degraded performance or mis-recognitions of triggering events. Future work will also attempt to study these techniques when applied to wearable devices with different form factors or located on different body locations beyond the wrist (e.g., necklace, belt) or for eyes-free input.

Conclusion

In conclusion, we have designed and implemented new techniques for augmenting normal touch input on a smartwatch to support pressure touch, twist and panning, which lead to many potential use cases. Our techniques use only the signal from integrated IMU to infer the action and its magnitude. We showed that it works on a variety of smartwatches in the market, via a software update. Our study showed our techniques are immediately feasible, with users' input accuracies in excess of 98.4%.

REFERENCES

1. Android. 2016. Rotation Vector. (2016). `http://developer.android.com/guide/topics/sensors/sensors_motion.html#sensors-motion-rotate`.
2. Apple. 2015. Apple Watch Force Touch. (2015). `http://www.apple.com/watch`.
3. Gierad Laput, Robert Xiao, Xiang 'Anthony' Chen, Scott E. Hudson, and Chris Harrison. 2014. Skin Buttons: Cheap, Small, Low-powered and Clickable Fixed-icon Laser Projectors. In *Proc. UIST '14*. ACM, New York, NY, USA, 389–394.
4. Masa Ogata and Michita Imai. 2015. SkinWatch: Skin Gesture Interaction for Smart Watch. In *Proc. AH '15*. ACM, New York, NY, USA, 21–24.
5. Jun Rekimoto and Carsten Schwesig. 2006. PreSenseII: Bi-directional Touch and Pressure Sensing Interactions with Tactile Feedback. In *Proc. CHI EA '06*. ACM, New York, NY, USA, 1253–1258.
6. Samsung. 2015. Samsung Gear S2. (2015). `http://www.samsung.com/global/galaxy/gear-s2/`.
7. Katie A. Siek, Yvonne Rogers, and Kay H. Connelly. 2005. Fat Finger Worries: How Older and Younger Users Physically Interact with PDAs. In *Proceedings of the 2005 IFIP TC13 International Conference on Human-Computer Interaction (INTERACT'05)*. Springer-Verlag, Berlin, Heidelberg, 267–280.
8. Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. 2007. Gestures Without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes. In *Proc. UIST '07*. ACM, New York, NY, USA, 159–168.
9. Haijun Xia, Tovi Grossman, and George Fitzmaurice. 2015. NanoStylus: Enhancing Input on Ultra-Small Displays with a Finger-Mounted Stylus. In *Proc. UIST '15*. ACM, New York, NY, USA, 447–456.
10. Robert Xiao, Gierad Laput, and Chris Harrison. 2014. Expanding the Input Expressivity of Smartwatches with Mechanical Pan, Twist, Tilt and Click. In *Proc. CHI '14*. ACM, New York, NY, USA, 193–196.
11. Hui-Shyong Yeo, Juyoung Lee, Andrea Bianchi, and Aaron Quigley. WatchMI: Pressure Touch, Twist and Pan Gesture Input on Unmodified Smartwatches. In *Proc. MobileHCI'16*. ACM.