WatchOut: Extending Interactions on a Smartwatch with Inertial Sensing

Cheng Zhang
Georgia Institute of Technology
chengzhang@gatech.edu

Junrui Yang
Peking University
me@jackieyang.me

Caleb Southern
Georgia Institute of Technology
caleb.southern@gatech.edu

Thad E. Starner
Georgia Institute of Technology
thad@gatech.edu

Gregory D. Abowd
Georgia Institute of Technology
abowd@gatech.edu

ABSTRACT

Current interactions on a smartwatch are generally limited to a tiny touchscreen, physical buttons or knobs, and speech. We present WatchOut, a suite of interaction techniques that includes three families of tap and swipe gestures which extend input modalities to the watch's case, bezel, and band. We describe the implementation of a user-independent gesture recognition pipeline based on data from the watch's embedded inertial sensors. In a study with 12 participants using both a round- and square-screen watch, the average gesture classification accuracies ranged from 88.7% to 99.4%. We demonstrate applications of this richer interaction capability, and discuss the strengths, limitations, and future potential for this work.

Author Keywords

Smartwatch; mobile interactions; inertial sensing; machine learning;

ACM Classification Keywords

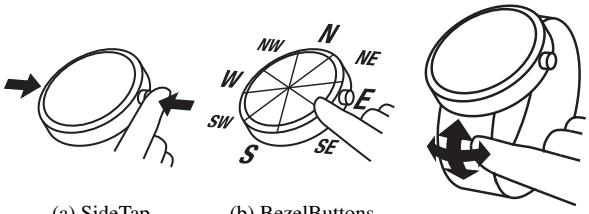
H.5.m. Information Interfaces and Presentation : User interfaces; Input devices and strategies

INTRODUCTION

The richness of touchscreen interactions on a smartphone cannot be easily replicated on a smartwatch. The relatively tiny screen exaggerates issues that already existed with smartphone interactions, such as the fat finger problem and occlusion [21]. There is even greater motivation to explore interaction techniques away from the touchscreen for these wrist-mounted devices.

We present WatchOut, using inertial sensors to provide a variety of tapping and sliding gestures on the side, bezel, and band of a smartwatch. WatchOut offers the following research contributions :

- The design and implementation of three gesture families that extend smartwatch interactions to the side, bezel, and band of the watch.
- An evaluation of the interaction performance with user-independent machine learning models in a laboratory environment.
- A demonstration of applications of WatchOut, and the practical challenges to improving and deploying the WatchOut interactions in real-world scenarios.



(a) SideTap (b) BezelButtons (c) BandSwipe

Figure 1: WatchOut Interaction Families

RELATED WORK

Novel interactions on mobile phones

Smartphones and wearable devices share many common challenges of mobile interaction design. Since Hinckley et al. first demonstrated the possibility of extending interactions on the phone in 2001 [8], much research has explored novel mobile interaction techniques by designing novel gestures on touchscreen [6, 15] or beyond [23, 12, 2] the touchscreen.

Novel interactions to support wearable devices

Compared with smartphones, the input for wearable devices is even more limited. To improve the input experience on wearable devices, researchers have proposed various novel input techniques. These involve the user wearing additional

ISWC'16, September 12-16, 2016, Heidelberg, Germany

DOI: http://dx.doi.org/10.1145/2971763.2971775

devices on the hands or wrist and arms [19, 4, 5] to detect hand gestures.

Novel interactions on smartwatches

As the smartwatch has become more popular, HCI researchers have explored how to improve the user experience in spite of the inherent limits of the small touchscreen and form factor. Approaches include increasing the size of the screen area [14], reducing the size of the touch area [21], or by applying carefully designed touchscreen gestures [16, 22]. Another approach is to make other parts of the watch interactive [1, 3], including the band[18], or an additional arm band [10]. Others have considered extending the interaction area to a larger space around the device; for example, recognizing 3D gestures in the space above [11], around [13, 17] the watch or expand the perception space by using dynamic peephole[9].

Most of these solutions require either some additional sensing infrastructure on the device or a completely new device to be worn by the user. Research [15, 23, 24] have shown the possibility of detecting the taps on the side and back of a phone with built-in sensors. However, the shape, the worn position, and the size of a watch is different from a smartphone. Therefore, how to design the gesture families and situated them into the watch applications are the new challenges. Compared with recent work [20] that recognizes the finger gestures by using built-in sensors on a smartwatch, WatchOut will demonstrate a comparable variety of input gestures on the watch case, with the advantage of using only existing inertial sensing common on smartwatches today.

GESTURE DESIGN AND DETECTION

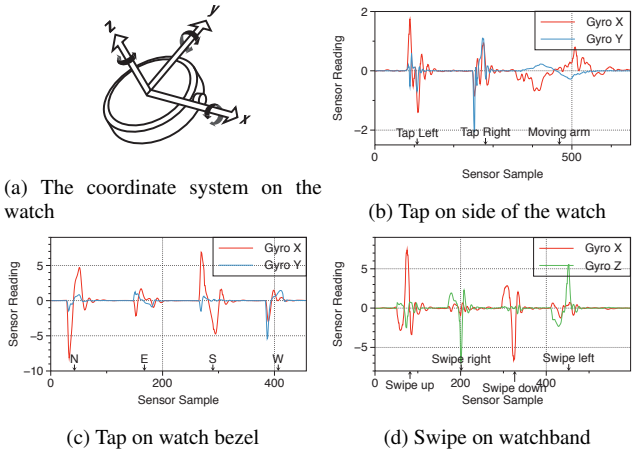
Smartwatches today commonly include two inertial sensors, an accelerometer and a gyroscope. We describe the response of these sensors to various physical stimuli and then define three families of interaction gestures based on these stimuli. Finally, we describe the data processing pipeline for gesture recognition.

Theory of Operation

In figure 2, we show the raw gyroscope data generated by tapping on the top of the watch bezel, the side of the watch bezel (what we call the case), and swiping on the watchband. We also observed similar patterns in the accelerometer data.

In figure 2b, we can see that a tap on the left will generate a positive spike along the x-axis of the gyroscope data, while a tap on the right will generate a negative spike. The lower frequency and lower intensity sensor data for an arm movement is also visually distinct from the harsh and high-frequency data from a tap gesture.

We also observed similarly clear data from taps on the top face of the watch bezel. Figure 2c shows four taps performed on each side of the watch bezel face (North, East, South, and West). Taps in the North and South will have relative larger x-axis gyroscope readings, and taps in the East and West will have a relative larger y-axis readings. Interestingly, the reading of the y-axis gyroscope generally appear to be smaller than the readings on the other two axes. This is because the



(a) The coordinate system on the watch

(b) Tap on side of the watch

(c) Tap on watch bezel

(d) Swipe on watchband

Figure 2: Sensor data collected when various gestures are performed

watch has more freedom to move along the x-axis due to the way a watch is worn.

Swipes on the watchband can also be detected through the gyroscope data (Figure 2d). Swiping up and down will tilt the watch positively and negatively along the y-axis, and swiping left and right will tilt the watch positively and negatively along the z-axis. These four gestures are visually apparent when comparing the strength of the signal for each axis and and the polarity of the peak.

Gesture Families

Based on our observations of the raw sensor data from various tap and swipe gestures around the case and band of a watch, we designed three families of gestures: SideTap; BezelButtons; and BandSwipe (see Figure 1).

SideTap

The user can tap on either the left or the right side of the watch case. These interactions can be performed eyes-free, and are appropriate for performing simple acknowledge or dismiss actions, such as rejecting a phone call.

BezelButtons

A user can tap the bezel area around the outside of the screen, as Figure 1b shows. We can recognize up to eight tap locations, whose positions are equally distributed around the watch case. Intuitively, these eight tap gestures can be used for navigating directionally, as with a D-pad. Potentially, BezelButtons can also help facilitate richer menus on the watch. For instance, most watch applications can only display a limited number of menu choices (usually around three items) due to the limited screen real estate.

BandSwipe

We can turn the band of a watch into an interactive surface by recognizing four different directions (up, down, left, right) of a sliding gesture, as Figure 1c shows. These four gestures can be naturally used in applications that require directional

controls. The BandSwipe gestures can also be used in combination with the touchscreen to enhance interaction.

Data Processing

To recognize a gesture, we first segment the sensor data for event detection, and then classify the gesture with pre-built machine learning models. We use the sequential minimal optimization (SMO) implementation of support vector machine (SVM) provided by Weka[7] to build the models, which are also used for real-time event detection and classification in our interactive prototype.

To recognize the gestures, we built two SVM models. The first model identifies sensor data as one of two classes: gesture or noise. If the data is classified as a gesture event, it is passed to the second model for classifying which gesture it is.

Event Detection

We collect gyroscope and accelerometer (linear acceleration) data on the watch at their highest sample rates.To process the data, we apply a one-second sliding window with 50% overlap. Within the window, we test the maximum absolute value for each axis of the gyroscope and the linear acceleration, respectively, against a threshold value. The thresholds are slightly different for each gesture family. If none of the threshold criteria are met, we identify that no gesture has been performed. If one of the threshold criteria is satisfied, we proceed to the next step. This threshold-based method is simple but has been proven effective in excluding non-gesture events in the preliminary experiments, especially when the wrist is still. However, too strict of criteria would increase false negative errors. We elected to set very loose thresholds, which can prevent false negative errors but allow false positive errors, which can be eliminated b classification later on.

If the above threshold criteria are passed, we identify the peak absolute value in the entire data buffer for each sensor (gyroscope and accelerometer), and the axis on which that peak falls. We use 0.25 seconds data before and 0.25 seconds data after the located peak to form an instance containing data of 0.5 seconds, which will be used for feature extraction. If the peak is too close to the end of the data buffer, we advance to the next one-second window with 50% overlap.

Feature Extraction

We first derive virtual sensors (Deri-gyro, Deri-linaccl) by calculating the derivative for each axis of the gyroscope and the linear accelerometer. For each sensor and virtual sensor axis, we calculate a set of statistical features—the minimum, maximum, mean, standard deviation, median, root-mean-square (RMS), variance, zero-crossing-counts, the values and difference of the first and second peaks. To capture the relative relationship between three axes of each sensor, we also calculate the ratio of the energy, the first and second absolute peak values, the difference between the first and second absolute peak values, and the correlation for the data of each two axes on each sensor. In total, 192 features are extracted for each half-second instance.

Classification

The features are used to train two support vector machines (SVM) for gesture classification. The first classifier distinguishes the noise instances from the gesture instances. If a gesture is detected, the second classifier then identifies which gesture is being performed and we move to the next window without overlap.

Choice of Hardware

We implemented our interaction techniques on two Android Wear smartwatches: 1) the LG G Watch Urbane; and 2) the Sony Smartwatch 3. We chose these two watches because they exhibit different physical characteristics. The LG Watch has a round screen and a leather watchband. The Sony watch has a square screen and a rubber watchband. We offload most computation work to a paired Google Nexus 6 smartphone, transmitting all of the sensor data wirelessly. In the future, it is conceivable that smartwatches will have sufficient computing power to run the gesture recognition software locally. All the inertial sensors are set to sample at the highest rate (200-250 Hz) during the data collection process.

Training the model

We collected training data from seven people to build two classification models (event detection and gesture classification) for each of the three gesture families and for each watch ($2 * 3 * 2 = 12 models$). The first three trainers were three of the authors. Each trainer provided 20-40 samples for each gesture, in order to train the gesture classifier. Each trainer also provided 20-40 samples of noise (non-gestures), in order to train the event detection classifier. The noise samples included actions such as raising the arm and rotating the wrist. After collecting these samples, we built the event detection model by labeling all the gestures as *gesture*, and the non-gestures as *noise*. For the gesture classification model, each gesture sample was labeled with its gesture name.

EVALUATION

We conducted a user study to to determine the extent to which a user-independent model can successfully select and classify the various input gestures. In addition, we report the impact of adding additional training data to the overall accuracy of a user-independent model for each gesture family on each watch.

Study Design

We evaluated WatchOut in a laboratory environment in order to determine how accurately our system could recognize unique gestures. We had each participant perform gestures from the three gesture families (SideTap, BezelButtons, and BandSwipe) on each of the two smartwatches (LG and Sony). We also collected survey responses before and after the interaction portion of the study.

Participants We recruited 12 participants (4 females, average age 25.7), all from a university campus. Three of the participants are daily smartwatch wearers, four had experience using a smartwatch, eight had never interacted with a smartwatch.

Procedure We asked participants to wear each watch on the left arm and use the right hand to perform the gestures. In

addition, we asked participants to wear the watch below the wrist joint on their arm, and to adjust the watch band so it fit tightly on the arm.

For each gesture family, we first demonstrated the interaction to each participant and then let them practice it. The participants were instructed to perform the gesture not too gentle by comfortably. The order of the two watches and the three gesture families for each watch were randomized. We then asked the participants to perform two sessions for each gesture family on each watch. Each session consisted of ten stimuli for each gesture in the family, presented in random order. The task was to perform the gesture indicated by each stimulus. In addition, participants also recorded two sessions of ten "noise" events by lifting their arms, as described in *Training* above. Each participant session lasted approximately one hour. For each stimulus presented to a participant, we saved the sensor data for the gesture in files with a length of 3 seconds for later analysis.

Results

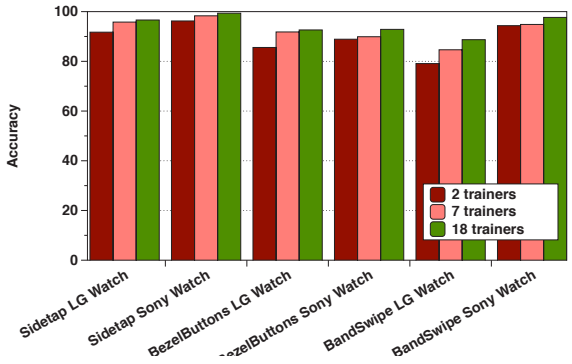


Figure 3: Overall accuracy of gesture classification for 12 participants, based on 2, 7, and 18 people in training set.

In total, we collected 6698 gesture samples from 12 participants. We ran several analyses of the gesture recognition performance offline, with data collected from the 12 participants and the training sets.

	False Positive Rate	False Negative Rate
SideTap_Round	0.63%	0.21%
SideTap_Square	0.21%	0.00%
BandSwipe_Round	2.40%	0.42%
BandSwipe_Square	0.73%	0.63%
BezelButtons_Round	0.16%	1.05%
BezelButtons_Square	0.00%	1.30%

Table 1: Overall event detection accuracy for each gesture family and watch.

Overall classification accuracy

In the first analysis, we used "leave-one-participant-out" methods. Both the noise-classifiers and gesture-classifiers were trained with all the data from the seven trainers, in addition to data from the remaining eleven participants.

The first stage of our gesture recognition pipeline is event detection. If we did not detect any gesture event in the sensor data from the files representing one gesture instance, we counted that instance as a false negative error. If we detect more than one gesture instance from the sensor data, we counted that instance as a false positive error. The false negative and positive instances (see Table 1 for rates) were not passed into the gesture recognition classifier.

Out of 6698 total gestures recorded, the total number of false positives was 37 and the total number of false negatives was 56. The false positive rates ranged from 0.00% (BezelButtons on the square watch) to 2.40% (BandSwipe on the round watch). The false negative rates ranged from 0.00% (SideTap on the square watch) to 1.30% (BezelButtons on the square watch).

The second stage of the pipeline is the gesture recognition classifier. The overall accuracy for gesture classification ranged from 88.7% for BandSwipe on the round watch/leather band to 99.4% for SideTap on the square watch (see the third/green bar for each gesture/watch condition in Figure 3).

We also produced a confusion matrix for all the gestures in each gesture family on each watch. The rows are the stimuli, and the columns are the recognized gestures. The accuracies of SideTap gesture recognition were between 96% to 100%, with the worst confusion of 4% on the round watch, mistaking a left tap for a right tap.

The confusion matrix for the BandSwipe gesture family is presented in Table 2a (round watch) and Table 2b (square watch). The gesture classification on the square watch was more accurate, with the worst confusion at 5%, mistaking an up swipe for a down swipe. There were more classification errors for BandSwipe on the round watch, with accuracies for each gesture ranging from 83% to 91%. We revisit the potential impact of the leather watch band on accuracy in the *Discussion* below.

Training with fewer data sets

Figure 3 also shows the results when the training model includes fewer examples. We compare the recognition results when only 2 trainers are used (leftmost/red bar for each condition), when 7 trainers are used (middle/pink bar), and when 18 trainers are used (rightmost/green bar). Figure 3 reveals that for all cases, the accuracy improves with more training data, but the increase is much greater when moving from 2 to 7 trainers than when moving from 7 to 18 trainers.

	↑	→	↓	←
↑	0.83	0.01	0.13	0.03
→	0.01	0.91	0.05	0.03
↓	0.05	0.00	0.91	0.03
←	0.04	0.02	0.04	0.89

(a) On the round watch.

	↑	→	↓	←
↑	0.95	0.00	0.05	0.00
→	0.00	0.99	0.00	0.01
↓	0.02	0.00	0.98	0.00
←	0.01	0.00	0.00	0.99

(b) On the square watch.

Table 2: Confusion matrix for BandSwipe

DISCUSSION

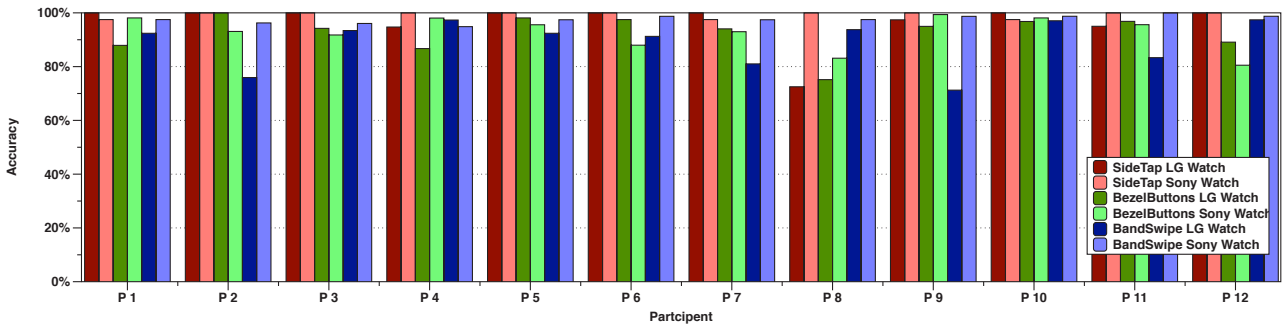


Figure 4: Gesture recognition accuracy for each participant

Most of the gestures were recognized with accuracies of above 90%, based on the user-independent model trained with data from 18 people. The best accuracy was observed for the SideTap gesture family, which is not surprising because it only included two gestures (left and right taps). The worst accuracy was for BandTap on the round watch, which had a leather band with stitching. The texture of this watch band, as opposed to the rubber band on the square watch, appeared to be more challenging for our gesture recognizer. We further discuss our results and explore the limitations of our approach before we demonstrate practical applications of the WatchOut gestures.

Generalizability

We evaluated WatchOut on both a round-faced and a square-faced watch. We built separate gesture recognition models for each watch, and for each gesture family. The overall accuracy for gesture recognition was similar for each watch on the SideTap and BezelButtons gesture families. In order to determine if the gesture recognition models were *device*-independent, as well as *user*-independent, we tested participants' data from the round watch with the models generated for the square watch, and vice versa (see Figure 5).

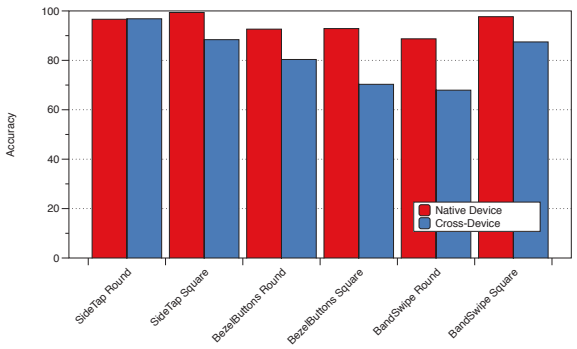


Figure 5: Device-independent gesture recognition accuracy.

The cross-device accuracy results for SideTap were virtually the same for the round watch (96.8% vs. 96.6% for the native model), and reasonably good for the square watch (88.4% vs. 99.4% for the native model). Because the SideTap model only needs to classify two distinct gestures, it is not suprising that these models exhibit the highest device independence. We suspect that the round watch model performed worse on the square watch, as opposed to vice versa, because the square watch better affords a tap consistently in the center of the left or right side as compared with the more ambiguous left or right target on a circle.

The BezelButtons gesture family does not appear promising for building device-independent models, at least between round and square watches. The cross-device accuracy results for BezelButtons were slightly worse for the round watch (80.4% vs. 92.6% for the native model), and considerably worse for the square watch (70.3% vs. 92.9% for the native model). We suspect that the distinct corners of the square watch provide a more specific and consistent target for the NE, SE, SW, and NW gestures as compared with the round watch. The different shapes of the watch cases may also produce different physical responses in the inertial sensors to taps on the bezel. In future work, we will investigate device-independent models between two different square watches, and between two different round watches, which could arise based on different relative locations of the inertial measurement units on those devices.

We observed the highest discrepancy of accuracy between watches for BandSwipe, likely based on the differences in each watch band. The square Sony watch had a smooth rubber band, while the round LG watch had a more textured, leather band with raised stitching. Overall, the BandSwipe recognition accuracy was considerably lower (88.7%) on the leather band as compared to the accuracy of 97.7% on the rubber band. As can be seen from Figure 4, the recognition accuracy for four participants (P2, P7, P9, P11) was particularly poor, each being less than 84%. These difference may be due to how the friction from swipes in the two materials is reflected in the inertial sensors, and also due to the difference described above on how some users performed swipes on the leather band. Therefore, to apply BandSwipe on a watch for achieving best accuracy, we suggest using the band which is made of high friction material (e.g. rubber).

BezelButtons Accuracy with Four or Eight Gestures

The BezelButtons family includes eight distinct gestures that our classifier can distinguish, as compared to four for BandSwipe and two for SideTap. The recognition accuracy was sufficiently high, at 92.6% for the round watch and 92.9% for the square watch. Upon examination of the confusion matrices, we discovered that the vast majority of classification errors were for adjacent gestures (e.g., NE or NW when N was intended). For the square watch (Table 3b), there was no confusion greater than 0.05% between any two non-adjacent gestures. For the round watch (Table 3a), there were only three instances of confusion between non-adjacent gestures, and these were at error rates of only 1%.

There are arguably more scenarios for which a family of four, as opposed to eight, distinct gestures could be useful. This is especially true due to the natural mapping of the four gestures as a direction-pad (N, E, S, W). To explore BezelButtons with four gestures (BezelButtons-4), we ran four additional analyses. We built new models that included only the N, E, S, and W gestures (BezelButtons-4N) for each of the two watches. We also built new models that included only the diagonal gestures (NE, SE, SW, SW) for both watches (BezelButtons-4NE).

The overall gesture recognition accuracy for BezelButtons-4 improved further on both watches, as compared with the accuracy for BezelButtons-8 (see Figure 4). On the round watch, the gesture recognition accuracy jumped to 98.3% and 99.3% , respectively for BezelButtons-4NE and BezelButtons-4N, as compared with 92.6% accuracy for BezelButtons-8. On the square watch, the gesture recognition accuracy jumped to 99.4% and 99.2%, respectively for BezelButtons-4NE and BezelButtons-4N, as compared with 92.9% accuracy for BezelButtons-8.

The confusion matrices for BezelButtons-4 appear in Figure 4. All individual gestures were recognized with at least 97% accuracy, and the only confusion that occurred was between adjacent gestures (e.g. NE recognized as NW). Although the recognition accuracies are sufficiently high for all of the BezelButtons-4 configurations, the N, E, S, W family is arguably a more natural mapping than the diagonal family. On the other hand, the four corners on the square watch provide distinct tapping targets for tasks such as a menu selection, but are not as appropriate for directional interactions.

Tightness and Position on the Wrist

We observed that many people usually wear the watch in one of two locations on the wrist. Some put the watch closer to the wrist, while others prefer to keep the watch a bit further down from the wrist joint, which gives the wrist more flexibility while in motion. In our study, we only tested our technique for the second position, which was well accepted by the participants. The majority of participants we observed wore the watch in the second position. Additionally, if the watch is worn in the first position, the bone on the joint of the wrist would be covered by the band, which limits the physical response of the inertial sensors to a tap or swipe event. In future work, we will explore a new model for the first wrist position with additional training data. In addition, we only tested the system when the watch was worn on the left wrist. In future work, we will explore how our model works with a watch on the right wrist, and determine if that requires separate training data.

In our study, we also asked the participants to adjust the watchband to fit tightly to the wrist. We are aware that some users prefer to wear the watch loosely, in which case, the watch may slide from the lower arm to the wrist back and forth while in motion, and thus change the nature of the inertial sensor data stream. A loosely worn style would introduce some recognition errors in our system, which is one of the limitations of our technique.

For some people, the comfort of the watch may be an impediment to use of watch gestures such as the WatchOut family in real-world scenarios. For example, P3 reported that the Sony square watch was heating up and the session was too long. Interestingly, P3's results were among the most accurate across all conditions, ranging from 91.8% to 100.0%.

Personalization

We evaluated recognition performance with user-independent machine learning models. Though our overall performance was very good, we did notice poorer performance for some participants (see Figure 4). One participant commented on getting tired during the lab experiment and switched to using his thumb to perform the gestures. Another participant had a particularly thin wrist and the watchband could not be adjusted as firmly as we would have wanted. For these users, we could consider building user-dependent models for the gesture classification. We would most likely pursue an incremental approach, whereby the user-dependent model could be augmented with training examples from an individual user, as demonstrated by Zhang *et al.* for BeyondTouch [23]. Considering only the participant sessions where overall accuracy was less than 84% (8 sessions across 6 unique participants), we tested how training data from that participant's session would impact classification accuracy. We gradually added the first 0,2,4,8,10 instances of each gesture into the training set and used the rest for testing. Across all of these 8 sessions, accuracy increased from an average of 77.9% to 86.0% as shown in figure 6.

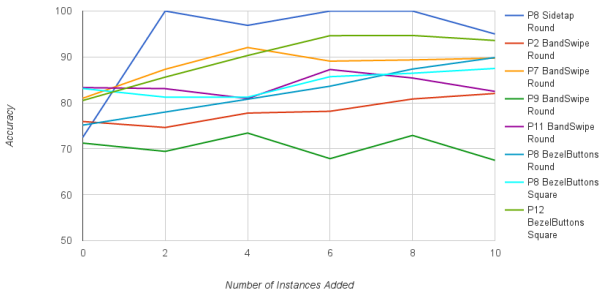


Figure 6: Impact of personalized learning models on classification accuracy

	N	NE	E	SE	S	SW	W	NW
N	0.90	0.01	0.00	0.00	0.00	0.00	0.00	0.09
NE	0.00	0.96	0.03	0.00	0.00	0.00	0.00	0.01
E	0.00	0.01	0.91	0.08	0.00	0.00	0.00	0.00
SE	0.00	0.00	0.03	0.93	0.04	0.00	0.00	0.00
S	0.00	0.00	0.00	0.04	0.92	0.04	0.00	0.00
SW	0.00	0.00	0.00	0.01	0.05	0.92	0.02	0.00
W	0.00	0.00	0.00	0.00	0.00	0.03	0.96	0.00
NW	0.04	0.01	0.00	0.00	0.00	0.00	0.03	0.92

(a) On the round watch.

	N	NE	E	SE	S	SW	W	NW
N	0.91	0.07	0.00	0.00	0.00	0.00	0.00	0.02
NE	0.00	0.91	0.08	0.00	0.00	0.00	0.00	0.00
E	0.00	0.03	0.94	0.03	0.00	0.00	0.00	0.00
SE	0.00	0.00	0.01	0.97	0.02	0.00	0.00	0.00
S	0.00	0.00	0.00	0.04	0.95	0.00	0.00	0.00
SW	0.00	0.00	0.00	0.00	0.01	0.98	0.00	0.00
W	0.00	0.00	0.00	0.00	0.00	0.04	0.87	0.08
NW	0.01	0.00	0.00	0.00	0.00	0.00	0.10	0.88

(b) On the square watch.

Table 3: Confusion Matrix for BezelButtons

	N	E	S	W
N	1.00	0.00	0.00	0.00
E	0.00	1.00	0.00	0.00
S	0.00	0.00	1.00	0.00
W	0.00	0.00	0.02	0.98

(a) BezelButtons-4N (Round)

	N	E	S	W
N	0.98	0.02	0.00	0.00
E	0.00	1.00	0.00	0.00
S	0.00	0.00	1.00	0.00
W	0.00	0.00	0.01	0.99

(b) BezelButtons-4N (Square)

	NE	SE	SW	NW
NE	0.97	0.00	0.00	0.03
SE	0.01	0.99	0.00	0.00
SW	0.00	0.01	0.99	0.00
NW	0.02	0.00	0.00	0.98

(c) BezelButtons-4NE (Round)

	NE	SE	SW	NW
NE	0.99	0.00	0.00	0.01
SE	0.00	1.00	0.00	0.00
SW	0.00	0.00	1.00	0.00
NW	0.00	0.00	0.01	0.99

(d) BezelButtons-4NE (Square)

Table 4: Confusion matrices for variants of 4-gesture Bezel-Buttons on square and round devices

Applications

We have designed and implemented applications that demonstrate how WatchOut can improve smartwatch interactions.SideTap offers two gestures, which we demonstrate acknowledging or dismissing an alert, such as a phone call. BezelButtons offers up to eight distinct gestures, which we demonstrate in an always-available interface for shortcuts to launch applications. BandSwipe offers an additional modality for navigation, which we demonstrate in a map navigation application.

We imagine extending these gesture families to other square, round, and band-type wearable surfaces: such as belt buckles, straps, and jewelry, in future work.

PRACTICAL CHALLENGES AND FUTURE WORK

Currently, our current technology was only evaluated in the lab-based environment and the noise data was collected by moving the arms and the wrist. Apparently, there are more challenges we need to address before deploying this technique in real applications. First, we plan to collect more data from watch wearers during the whole day and evaluated our system in the real applications in the future. Second, WatchOut provides three sets of input gestures, which was only evaluated separately. In real applications, the user may prefer to use these three gestures sets interchangeably. How to switch modalities is another potential challenge. One solution is to link each gesture set with a separate application. Therefore, the certain gesture set will only be detected when the associated application starts. In addition, we also plan to explore how to detect the context which could be used to automatically switch gesture set. For instance, the sound generated by different gesture sets are different, which could be used to determine which gesture set should be detected.

CONCLUSION

We have presented WatchOut, a suite of interaction techniques that extend the input modalities on a smartwatch. WatchOut employs a user-independent gesture recognizer pipeline, based on data from the accelerometer and gyroscope embedded in a watch, which can distinguish up to eight distinct gestures. In a study with twelve participants using both a round- and square-faced watch, we demonstrated high gesture recognition accuracy of 88.7% to 99.4%. We then presented interactive demonstration applications for the SideTap, BezelButtons, and BandSwipe gesture families, and discussed the advantages and limitations of using the WatchOut gestures in real-world scenarios.

ACKNOWLEDGMENTS

We would like to thank Ning Xu for the help on the picture and video, all their reviewers for their feedback and the participants in the user study. This work is partly supported by the Intel Science and Technology Center for Pervasive Computing (ISTC-PC).

REFERENCES

1. Ashbrook, D., Lyons, K., and Starner, T. An investigation into round touchscreen wristwatch interaction. In *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services - MobileHCI '08*, Association for Computing Machinery (ACM) (2008).
2. Baudisch, P., and Chu, G. Back-of-device interaction allows creating very small touch devices. In *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*, Association for Computing Machinery (ACM) (2009).
3. Blasko, G., and Feiner, S. An interaction system for watch computers using tactile guidance and bidirectional segmented strokes. In *Eighth International Symposium on Wearable Computers*,

4. Dementyev, A., and Paradiso, J. A. Wristflex: Low-power gesture input with wrist-worn pressure sensors. In *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*, Association for Computing Machinery (ACM) (2014).
5. Fukui, R., Watanabe, M., Shimosaka, M., and Sato, T. Hand shape classification in various pronation angles using a wearable wrist contour sensor. *Advanced Robotics 29*, 1 (jan 2015), 3–11.
6. Goel, M., Wobbrock, J., and Patel, S. Gripsense: Using built-in sensors to detect hand posture and pressure on commodity mobile phones. In *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*, Association for Computing Machinery (ACM) (2012).
7. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The weka data mining software: an update. *ACM SIGKDD explorations newsletter 11*, 1 (2009), 10–18.
8. Hinckley, K., and Horvitz, E. Toward more sensitive mobile phones. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, ACM (2001), 191–192.
9. Kerber, F., Krüger, A., and Löchtefeld, M. Investigating the effectiveness of peephole interaction for smartwatches in a map navigation task. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*, ACM (2014), 291–294.
10. Kerber, F., Lessel, P., and Krüger, A. Same-side hand interactions with arm-placed devices using emg. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, ACM (2015), 1367–1372.
11. Kim, J., He, J., Lyons, K., and Starner, T. The gesture watch: A wireless contact-free gesture based wrist interface. In *2007 11th IEEE International Symposium on Wearable Computers*, Institute of Electrical & Electronics Engineers (IEEE) (oct 2007).
12. Kratz, S., and Rohs, M. Hoverflow: expanding the design space of around-device interaction. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '09*, Association for Computing Machinery (ACM) (2009).
13. Laput, G., Xiao, R., Chen, X. '., Hudson, S. E., and Harrison, C. Skin buttons: cheap, small, low-powered and clickable fixed-icon laser projectors. In *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*, Association for Computing Machinery (ACM) (2014).
14. Lyons, K., Nguyen, D., Ashbrook, D., and White, S. Facet: a multi-segment wrist worn system. In *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*, Association for Computing Machinery (ACM) (2012).
15. McGrath, W., and Li, Y. Detecting tapping motion on the side of mobile devices by probabilistically combining hand postures. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, ACM (2014), 215–219.
16. Oakley, I., Lee, D., Islam, M. R., and Esteves, A. Beats: Tapping gestures for smart watches. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, Association for Computing Machinery (ACM) (2015).
17. Ogata, M., and Imai, M. Skinwatch: skin gesture interaction for smart watch. In *Proceedings of the 6th Augmented Human International Conference on - AH '15*, Association for Computing Machinery (ACM) (2015).
18. Perrault, S. T., Lecolinet, E., Eagan, J., and Guiard, Y. Watchit: simple gestures and eyes-free interaction for wristwatches and bracelets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*, Association for Computing Machinery (ACM) (2013).
19. Rekimoto, J. GestureWrist and GesturePad: unobtrusive wearable interaction devices. In *Proceedings Fifth International Symposium on Wearable Computers*,
20. Wen, H., Ramos Rojas, J., and Dey, A. K. Serendipity: Finger gesture recognition using an off-the-shelf smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM (2016), 3847–3851.
21. Xia, H., Grossman, T., and Fitzmaurice, G. Nanostylus: Enhancing input on ultra-small displays with a finger-mounted stylus. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology - UIST '15*, Association for Computing Machinery (ACM) (2015).
22. Xiao, R., Laput, G., and Harrison, C. Expanding the input expressivity of smartwatches with mechanical pan, twist, tilt and click. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, Association for Computing Machinery (ACM) (2014).
23. 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 *Proceedings of the 20th International Conference on Intelligent User Interfaces - IUI '15*, Association for Computing Machinery (ACM) (2015).
24. 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 *Proceedings of the Adjunct Publication of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13 Adjunct, ACM (New York, NY, USA, 2013), 111–112.