BlyncSync: Enabling Multimodal Smartwatch Gestures with Synchronous Touch and Blink

Bryan Wang
University of Toronto
Toronto, Ontario, Canada
bryanw@dgp.toronto.edu

Tovi Grossman
University of Toronto
Toronto, Ontario, Canada
tovi@dgp.toronto.edu

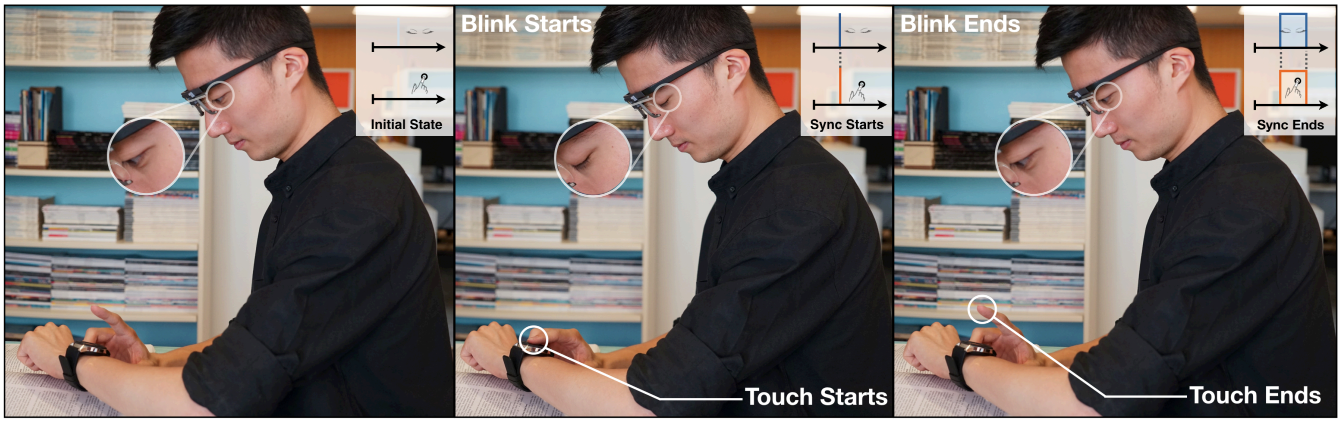

Figure 1. The BlyncSync gesture set is enabled by performing a pair of synchronous blink and touch events. The gesture is only recognized if the touch and blink events start and end within a small threshold, reducing the chance of false activations.

ABSTRACT

Input techniques have been drawing abiding attention along with the continual miniaturization of personal computers. In this paper, we present BlyncSync, a novel multi-modal gesture set that leverages the synchronicity of touch and blink events to augment the input vocabulary of smartwatches with a rapid gesture, while at the same time, offers a solution to the false activation problem of blink-based input. BlyncSync contributes the concept of a *mutual delimiter,* where two modalities are used to jointly delimit the intention of each other's input. A study shows that BlyncSync is 33% faster than using a baseline input delimiter (physical smartwatch button), with only 150ms in overhead cost compared to traditional touch events. Furthermore, our data indicates that the gesture can be tuned to elicit a true positive rate of 97% and a false positive rate of 1.68%.

Author Keywords

BlyncSync, Gaze UI, Smartwatch, Mutual Delimiter, Mobile HCI, Wearables.

CSS Concepts

• **Human-centered computing~Human computer interaction (HCI)~Interaction techniques**;

CHI '20, April 25–30, 2020, Honolulu, HI, USA.
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-6708-0/20/04...$15.00.
DOI: http://dx.doi.org/10.1145/3313831.3376132

INTRODUCTION

Input techniques have been drawing abiding attention along with the continual miniaturization of personal computers in the past decades. From tabletop computers to pocket-carried mobiles; wrist-placed smartwatches to ubiquitous IoT devices, HCI researchers strive to develop proper input modalities to interact with emerging devices with different form factors and using scenarios.

Oftentimes, multiple input channels are incorporated all together if a single channel is deficient to enable satisfactory interaction. For instance, though touch input is replacing mouse and keyboard as the dominant input modality as computing shifting from PCs to mobile devices, it becomes notoriously challenging when the form factors further diminish to the size of a smartwatch. Therefore, despite smartwatches are becoming increasingly popular, manufacturers still struggle to design self-sufficient input techniques providing adequate operations. As a result, commercial products often adopt hybrid methods combining multiple input modalities, such as touch screen, hardware buttons, bezels, and voice input.

Similarly, while eye inputs, e.g. gaze or blink, have been shown promising to provide instant and subtle interaction [14, 37, 59], it suffers from the problem of false activation as human eyes are meant to serve as perceptive organs, and did not evolve to be fully controllable – *the Midas Touch Problem* [23]. Therefore, additional mechanisms, such as dwell [23], smooth pursuit movements [14] or additional controllers [47] , need to be utilized to mitigate the problem of false positives.

These modalities, however, often either contribute independently to the input expressivity or only serve as auxiliary modalities supplementing the primary input channel. In contrast, inspired by the existing concept of *mutual disambiguation* [43], where two error-prone modalities are combined to reduce the recognition errors of each other, we see the opportunity of blending multiple input channels to delimit and reinforce each other. In this paper, we present the concept of a *mutual delimiter*, where two modalities are used to jointly delimit the intention of each other's input – therefore increasing their respective input vocabularies and/or addressing any potential issues of false activations. We instantiated the concept of *mutual delimiter* by designing BlyncSync, a novel multi-modal gesture set that leverages the synchronicity of touch and blink events to delimit both input at the same time. augment the input vocabulary of smartwatches. In doing so, BlyncSync improve the input vocabulary of smartwatch with a rapid gesture, while contributing a generalizable solution to the false activation problem of both blink-based and touch-based input.

To inform the design of BlyncSync, we first conducted a lab study collecting data of touch and blink patterns during smartwatch use. The analysis showed that it is rare for subjects to perform synchronous touch and blink events. We then conducted a second study to evaluate the BlyncSync gestures set. The results showed that BlyncSync is easy to perform and is 33% faster than using a baseline input delimiter (physical smartwatch button), with only 150ms in overhead cost compared to traditional touch events. Furthermore, our data shows the gesture can be tuned to elicit a true positive rate of 97% while achieving a false positive rate of 1.68%. We then demonstrate several application scenarios that utilize BlyncSync gestures and discuss future lines of work.

RELATED WORK

In this section, we review research leveraging synchronous and correlated signals as input, research that enhances the input of smartwatches, as well as research that uses the eye as an input channel.

Synchronous Gestures and Motion Correlation

Prior work has investigated the use of synchronous signals as device input. In general, synchrony could be achieved by synchronous signals from multiple sources, such as devices [32, 35] or users [50]. Hinckley proposed a synchronous gesture of bumping two devices together for dynamic display tiling [17]; Smart-Its Friends [12] and SyncTap [52] used synchronous input on two devices to establish network connections. Synchronous gestures have also been used to enhance smartwatch input [35, 53, 62, 67]. Relevantly, Velloso et al. presented motion correlation [57] where user input becomes an act of synchronizing with a displayed motion for target selection. In contrast to these systems, BlyncSync is the first to explore the use of two synchronous input signals (blink and touch) from a single user to mutually delimit and thereby reinforce both input modalities.

Augmenting Smartwatch Input

Mitigating the fat-finger problem on [20] tiny touch screens and augmenting the input space for smartwatches have both been well explored to expand the input expressivity of smartwatches. For example, ZoomBoard [42] and Swipeboard [11] use multi-stroke touch gestures to enable accurate text entry on tiny screens. NanoStylus [63] uses a finger-mounted stylus to avoid finger occlusion. TouchSense [21] leverages the differences in finger contact areas when touching at different angles.

Past research also explored extending the input area of smartwatches to other components, such as the band [36, 46], bezel [5, 45] edge [40] and face [64]. Interactions surrounding the watch have also been explored. Skin Buttons [31] use projected icons and infrared proximity sensors to extend smartwatch interaction onto the arm. Abracadabra [16] and Gesture Watch [26] use a magnetic ring and magnetometer to track in-air finger motion. HoverFlow [28] and AuraSense [69] enable similar interactions using infrared proximity sensors and electric field sensing.

Most of these prior techniques only exploit a single input modality (touch), leaving other input modalities unused. Furthermore, gesture-based techniques for any interactive platform are limited, in that they require an explicit delimiter, to distinguish between command gestures and default command input [18]. Our work leverages the additional input channel of eye blinks, creating new opportunities for multimodal interaction with smartwatches, while also introducing a novel delimiter for command input.

Eye Expression as Input Channel

Eye expression, such as gaze [14, 23, 47] and blink [27, 29, 37] has long been considered as an alternative input modality for applications and users that require hands-free control [6, 23, 37]. For example, Jacob studied utilizing fixated gaze input on statically presented targets for selection [23], whereas BlinkWrite [6, 37] uses blink as the only input modality to enter text on a scanning keyboard. However, since eye input is prone to involuntary activation, it usually requires additional mechanism, e.g. long dwell on the intended target, long blink, or supplementary buttons. Recent research [14, 60] has found using smooth pursuits, slow and consistent eye movements that occur when the eyes follow a moving object, can effectively prevent false activations. Particularly, Orbits [14] applied smooth pursuits on smartwatches to enable hands-free input. However, to reduce false positives, Orbits suggests a 1-second activation period to recognize gestures.

It has also been suggested in prior work that eye expression would be better served as an auxiliary input modality [47, 66] supplementing more traditional channels of input. For example, Zhai et al. [66] proposed to use gaze to improve the performance of manual pointing using gaze for distant jumps and then fine-tune the final position with manual cursor movements. Gaze-Touch [47] and Gaze-Shifting [48] further extends the concept to multi-touch and touch+pen devices.

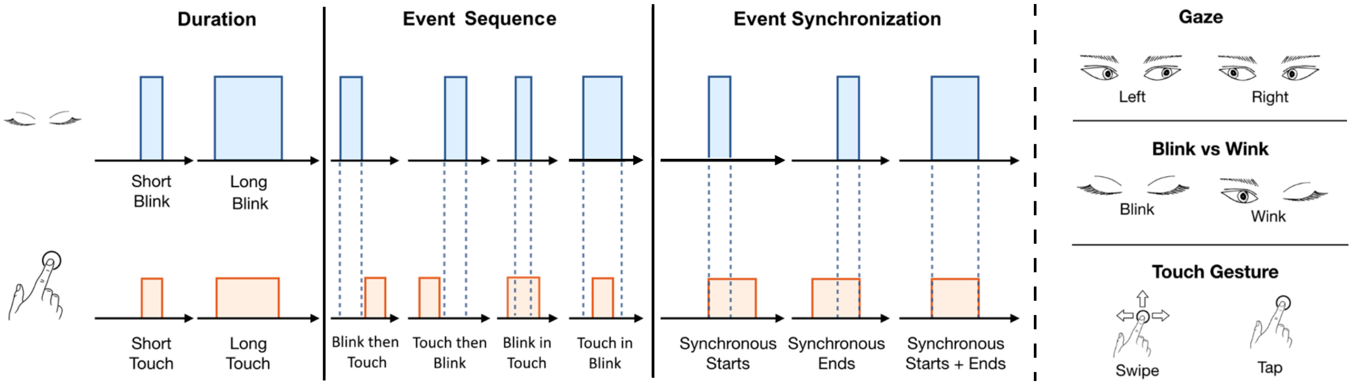

Figure 2. The first part of our Touch & Blink gesture design space relates to the duration and timing of the touch and blink events. The second part of our Touch & Blink gesture design space relates to the nature of the touch and blink events.

There is also a significant body of work exploiting the subtleness of gaze for private and secure interaction on mobile devices. For example, to prevent shoulder surfing when entering password, Khamis et al. [24, 25] use the combination of gaze and touch while Almoctar et al. [2] leverage both back-of-device touch and smooth pursuit eye movements. Eye modality is also used to enhance speech interaction [13, 68] since it helps relieve the need for the lengthy descriptions when specifying objects. Our work differs with the reviewed literature in two ways. First, we focus on the underexplored space that takes advantage of the synchronicity between eye and touch. Moreover, our work suggests a novel solution to the false activation problem for both touch gestures *and* eye-based input, by utilizing both modalities at the same time to create a "mutual delimiter".

DESIGN SPACE OF TOUCH & BLINK GESTURES

The key insight of our work is to design gestures based on the synchronous occurrence of touch and blink events, which opens a large set of possible input gestures. We define *Touch & Blink Gestures* as the broad class of gestures that consist of coordinated touch and blink events. Specifically, we define a touch as any gesture involving placing and removing fingers on a touchscreen, including tap, swipe, and long press, etc. A blink by definition is the process of closing and opening both eyes once. To the best of our knowledge, no previous work has considered this class of gestures. Therefore, we first discuss an associated design space grounded by prior literature in gesture input.

Duration

Threshold durations have been leveraged in prior work to differentiate deliberate blinks from natural ones [6, 37]. Similarly, extended touch events are widely used across commercial touch-based products to distinguish between different actions. While durations can improve a system's ability to distinguish between a user's intent, it has the trade-off of slowing down the pace of the user and interrupting the flow of interaction. As such, we avoid any restrictions on the duration of events required for BlyncSync.

Event Sequence

Just as a tap and double tap can have different meanings, it is possible to design Touch & Blink Gestures with multiple blinks and touches. More complex sequences using Morse code [39] or rhythm-based gestures [15, 41, 61] could also be used. However, as sequence lengths increase, so too will the complexity of the gestures. Therefore, our designs will focus on just a single touch and blink event.

Event Synchronization

Because touch and blink events can occur in parallel, it is important to understand and formally define what it means for two events to be synchronous. We propose to decompose each event into its start and end times (Figure 3), resulting in three types of synchronous Touch & Blink gestures: *synchronous start*, *synchronous end*, and *synchronous start and end*. Because it is impractical to expect humans to perform gestures with perfect synchronization, a threshold tolerance must be applied to account for offsets. Setting threshold offsets could be a key method for reducing false positives, but at the cost of increasing the required accuracy to perform the gestures. As such, this parameter will be tuned based on results from the first study on false activations.

Gaze

To enable Touch & Blink gestures, it is likely that eye tracking equipment will be used to detect the user's blinks. The gaze direction could thus serve as an additional enhancement to Touch & Blink Gestures. Our initial design will not leverage gaze but will demonstrate its use later in our application scenarios.

Blink vs Wink

Wink is another possible eye expression for interaction [38]. A benefit of using winks is that it will be robust to false activations, as naturally occurring winks are rare. However, winks are more prone to causing eye-fatigue, and some people are unable to wink altogether [44]. To avoid these additional difficulties, we utilize blink events only.

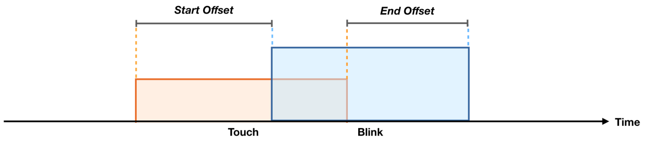

Figure 3. Event Synchronization. When comparing the two events, both the start and end offset can be considered.

Touch Gesture Type

A final consideration for Touch and Blink gestures is the nature of the touch gesture. The two most common forms of touch input events are tap and swipe (or drag). Both of these gestures can be quick and are easy to perform. Furthermore, swipe events can expand the gesture vocabulary by distinguishing between different directions [30]. As such, we consider both tap and swipe gestures in our explorations.

Summary

The design space of Touch & Blink Gestures produces many interaction possibilities. To narrow our scope, we have focused on gestures that will be easy and fast to perform. This is especially important for smartwatch designs, which are meant to enable short bursts of interaction [3, 4]. In our design of BlyncSync, only a subset of the design space was exploited. However, it is our hope that by presenting this broader design space, we can frame our design of BlyncSync within the broader class of Touch & Blink Gestures and inform further efforts in this area.

STUDY 1: FALSE ACTIVATIONS

Utilizing synchronization between touch and blink events could create a simple gesture set that will have low false activation rates. However, it is unclear how often synchronous touch and blink events would occur involuntarily. Previous studies have revealed average blink rates, but it has been found that blink rates are dependent on the context of a user's task [49]. In this study, we examine the patterns of touch and blink when participants are using two different smartwatch applications. We also include two analogous physical tasks, to compare to blink rates during similar paper-based tasks.

Participants

We recruited 12 participants (5 females) with a mean age of 26.1 (min=22, max=30). All the participants recruited had normal, uncorrected vision to avoid the interference that eyeglasses could have on blink detection.

Tasks and Procedure

The participants were asked to perform four tasks, two with a smartwatch and two with physical paper. We designed two smartwatch apps that had varying degrees of interaction intensity: *Calculator* and *News Feed*. For the *Calculator* task, which required intensive input, the application would show a formula for participants to type. Once the formula was correctly typed, the equal sign button would appear for participants to submit the calculation. Another formula would show after the answer was calculated. The *News Feed* task elicited lighter interaction intensity, requiring a mix of reading and touch input. Users would swipe to read through short news articles. For each article, the app would show a photo and a title. After clicking the title, the app would show the content of the news. Participants were asked to read the text, and then press a button to switch to the next article.

The paper tasks were *reading an article* and *solving a word search puzzle*. The former requires no hand action, eliciting lighter interaction, while the latter requires intense interaction with the hand.

Users were asked to perform each task for 5 minutes. While participants knew they were wearing eye-tracking hardware, they were not informed that blink rates were being tracked until the study was completed. The participants' heads were fixated with a Good-Lite chin rest and they rested their arm on a support stand during the study to reduce fatigue. Before the study, we adjusted the height of desk, chair and chin rest to ensure participants felt comfortable throughout the study.

Study Design

The study was a 2x2 within-subject design, with independent variables of *platform* (*smartwatch*, *paper*), and *intensity* (*light*, *heavy*). Each combination of independent variables mapped to one of the four tasks, as described above (Figure 4). The ordering of independent variables was fully counterbalanced. Participants were given optional one-minute breaks between tasks.

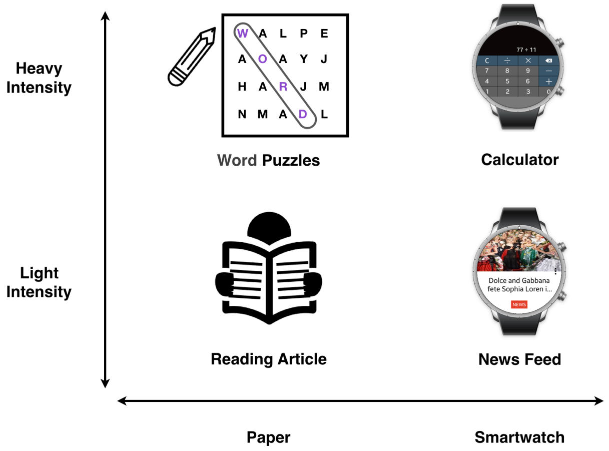

Figure 4. The 2x2 design of Study 1 and four associated tasks.

Implementation and Apparatus

We conducted the experiment in a quiet laboratory setting where participants sat at a height-adjustable desk wearing a Samsung Galaxy Watch 46MM (SM-R800), a multi-touch smartwatch with a circular AMOLED screen (360 x 360 resolution) running the Tizen Based Wearable OS 4.0. Participants' eyes were tracked by a Tobii Pro Glasses 2 with a tracking frequency of 50 Hz. As shown in Figure 5, a Python web server was developed and run on a Macintosh laptop. The eye tracker data was streamed to the server via a wireless UDP connection. For data transmission between the smartwatch and server, we used WebSocket due to its low latency and bi-directional communication. We applied Cristian's clock synchronization algorithm [12] to calibrate the time systems of the smartwatch, eye tracker and server. The smartwatch applications were written in JavaScript. We also recorded the study with a Logitech C922 Pro Stream HD Webcam with 60 fps at 720p, directed at the user's face.

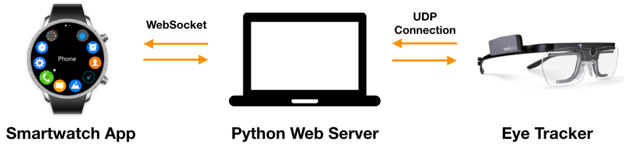

Figure 5. The system diagram.

		🟧		🟦	
		Rate	**Duration**	**Rate**	**Duration**
⌚					
📄	**Puzzle**	N/A	N/A	4.3/min (σ = 3.2)	62ms (σ = 9.9)
	Article	N/A	N/A	8.2/min (σ = 5.3)	70ms (σ = 12.8)

Table 1. Average blink and touch rates, and their durations.

Blink Detection

We leveraged the gaze data yielded by the eye tracker to detect the occurrence of a blink. Blinks and their durations were calculated during times when the pupil center-coordinates of both eyes were not reported. The approach leverages the specialized hardware and algorithms of the eye tracker and requires no additional computational cost. During manual review of the webcam footage, we observed some events were falsely classified as blinks if the user moved their eyes quickly. As such, we only classified a blink if the gaze velocity was under 6.25 units/s in the eye tracker's gaze point coordinate system. During pilot tests, we found our method was faster and more accurate than webcam-based OpenCV algorithms. To prevent the eyewear from dropping down the nose, we controlled the participant's view angle and ensured the glasses were comfortably positioned.

Results

Across the 12 participants, a total of 7067 touch events and 1562 blink events were collected. The average blink and touch rates, and their durations, are reported in Table 1. The task intensity had a significant effect of blink rates ($F_{1,11}$ = 12.235, $p < 0.05$), but there was not an effect of platform on blink rate. Consistent with prior literature, blink rates were lower when users were required to perform more intensive tasks [1, 65]. Blink durations were consistent among tasks except that of *solving a word search puzzle*, which was slightly shorter than others. This may have been due to a heavier visual workload [8]. Touch durations, as expected, were longer for the news feed task, which required dragging operations, while the calculator only required rapid tapping. The rates of blink and touch presented are based on a controlled lab study, we will discuss later in the paper how they might vary under different conditions.

False Activation Analysis

The main goal of the study was to understand how to design the BlyncSync gesture set in a way which minimizes false activations. Specifically, we were interested in knowing how often blink and touch events occur synchronously. As described in our design space, events could be considered synchronous if they start at the same time, end at the same time, or start *and* end at the same time. To analyze these variations, we took every touch event that occurred, and found its nearest blink event. We then calculated the distance between their start times and end times:

$$offset_{start} = start\_time_{touch} - start\_time_{closetblink}$$
$$offset_{end} = end\_time_{touch} - end\_time_{closestblink}$$

Figure 6 shows the results in a 2D dimensional space. The scatterplot is divided into four quadrants representing the sequential ordering: The bottom-left represents touches that occurred before a blink; the top-right represents touches that occurred after its closest blink; the top-left represent touches that completely encompassed a blink event; the bottom-right represents touch events that occur during a blink.

From the plot we can see that most data points are distributed in the bottom-left (51.74%) and top-right (46.67%) quadrants. This is reasonable since the other two quadrants represent an entire event contained by another. Given the short durations of both touch and blink events, this is unlikely to occur. All points in the top-left quadrant were swipes (1.58%), as blink events occurred within them. Only one point (0.01%) is in the fourth quadrant, representing touch events occurring within a blink. Across each quadrant, we consider the data points close to the origin representing a *synchronous* blink and touch. An offset of 250ms is used for illustration in Figure 6, representing 1.68% of all data.

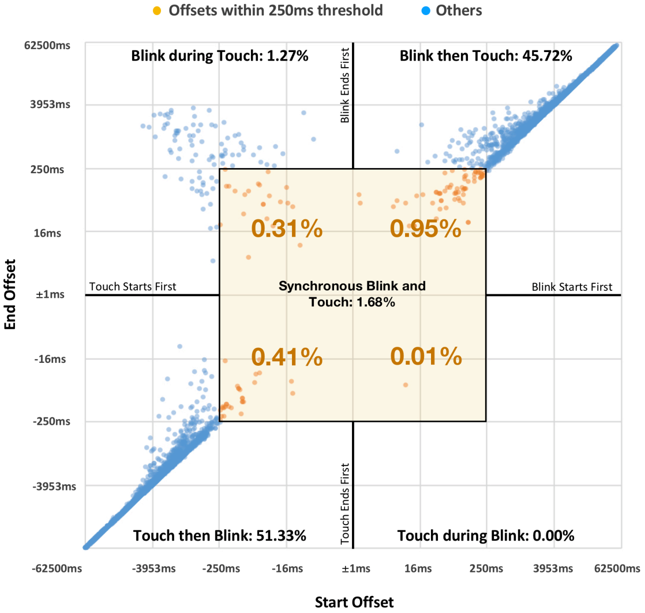

Figure 6. Offsets for all touch events (logarithmic scale). The majority of points fall within the top right and bottom left quadrants. Fewer than 2% of samples (orange dots) have a start and end offset both within 250ms (central yellow square).

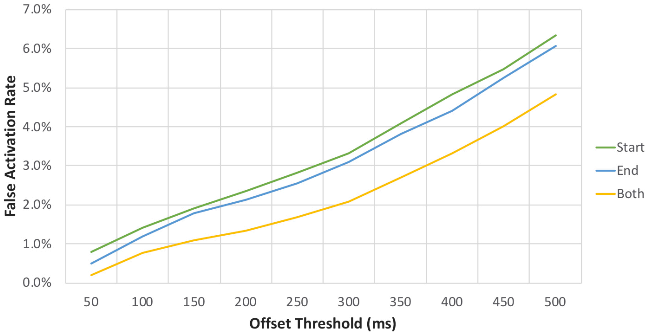

Figure 7. False activation rates by offset threshold when synchronizing the start, the end, and both the start and end.

The false activation rates for different offset values are shown in Figure 7. As illustrated, synchronizing both the start and end times results in lower false activation rates. The results are promising - only 0.2% of events had a start and end offset within 50ms, and only 2.7% of events had a start and end offset within 350ms. This data shows that by imposing an appropriate offset threshold, there should be an opportunity to design a Touch & Blink Gesture that will be easy to perform, while achieving low false activations.

BLYNCSYNC

Based on the results of study 1, we designed BlyncSync, a gesture set consisting of synchronous blink & touch events. To increase the gesture set vocabulary, we defined 5 gestures: a blink synchronized with either a tap (*BlyncSyncTap*), or a swipe in 4 directions (*BlyncSyncUp*, *BlyncSyncDown*, *BlyncSyncLeft*, *BlyncSyncRight*).

Guided by the results of our first study, we use both the start and end offset values for defining synchronization to minimize false activations. The Study 1 data indicates that a 50ms offset would achieve a false activation rate of 0.2%. However, smaller thresholds would increase the precision needed to perform the gesture. We piloted 2 users' ability to touch and blink at the same time and found they could typically do so within an average of 150ms. To allow for additional tolerance, we set the BlyncSync offset threshold to 250ms, which would still maintain a false positive rate of 1.68%, according to our Study 1 data.

BlyncSync Implementation

The software and hardware implementation of BlyncSync is identical to the setup as described in Study 1. This approach enables real-time recognition and visual feedback. To classify the gesture, the recognizer needs to wait up to 250ms for a subsequent blink after each touch event. However, to account for data transmission delays (roughly 50ms), we chose a larger pause (400ms), to ensure all events are received prior to classification. This delay does not affect the synchronization recognition, as the time systems of the eye tracker and smartwatch are calibrated. The event is classified by calculating the offsets between the received touch and closest blink event. If both events start and end within 250ms of one another the event is classified as a BlyncSync gesture.

STUDY 2: TECHNIQUE EVALUATION

We conducted a study to evaluate the BlyncSync gestures. The study had three main goals. First, we wanted to test the usability, accuracy, and efficiency of the BlyncSync gestures. Second, we wanted to determine an optimal balance between false positives and true positives, by examining the actual offsets users would produce when trying to perform simultaneous blink and touch gestures. Third, we wanted to compare BlyncSync to existing techniques.

For baseline comparison, we compared BlyncSync to using the physical button on the side of the smartwatch (*ButtonTouch*), a technique that is commonly utilized commercially to delimit between modes of interaction. We also compare BlyncSync to traditional unaltered touch gestures (*TouchOnly*). This condition was included in our study to understand the nature and extent of any overhead cost of the proposed technique. While we also considered other gaze-based techniques from the research literature (e.g. [14, 24, 25]), we elected not to include them in the study, as they were designed for different contexts. In our discussion section, we contrast our findings to prior techniques.

Participants

We recruited 12 participants (4 females) with a mean age of 25.2 (min=22, max=28). Participants had normal uncorrected vision. Participants were paid $20 for completing the study, which took approximately 50 minutes.

Tasks and Procedure

Participants performed 5 gestures (tap, swipe up, swipe down, swipe left, swipe right) using 3 different techniques (*TouchOnly*, *ButtonTouch, BlyncSync*). Participants were asked to perform each gesture using their index finger and were told to perform the gestures as fast and accurately as possible. To begin a trial, participants first tapped the middle of the screen, which would start the task timer. An instruction would then be displayed, consisting of an arrow for the four swipe directions, and a dot for the tap gesture. The trial would end once the gesture was completed (Figure 8).

For *TouchOnly*, the user could immediately perform the gesture upon receiving the instruction. For *ButtonTouch*, there would be a large rectangle behind the instruction to remind participants to first press the physical button before performing the gesture. After pressing the top right physical button of the watch, the rectangle would disappear, and the gesture could be performed. For *BlyncSync*, the instruction looked the same as for *TouchOnly,* but the user needed to blink while performing the touch gesture. The gesture was not considered complete until both the blink gesture event and touch event were completed (Figure 8).

After each trial, visual feedback was shown. If a gesture was performed incorrectly, the reason of failure (e.g. blinking too early, swiping in the wrong direction) would be displayed on the screen. To support natural use, users were not required to use the chin rest or hold their arm in a specific posture, as we found our system was still able to detect blink events reliably.

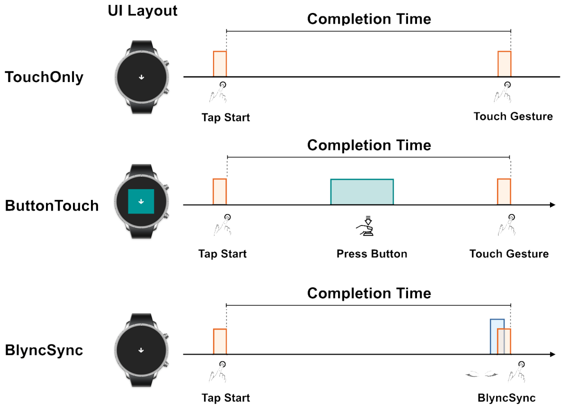

Figure 8. Task instructions and event sequences for TouchOnly, ButtonTouch, and BlyncSync.

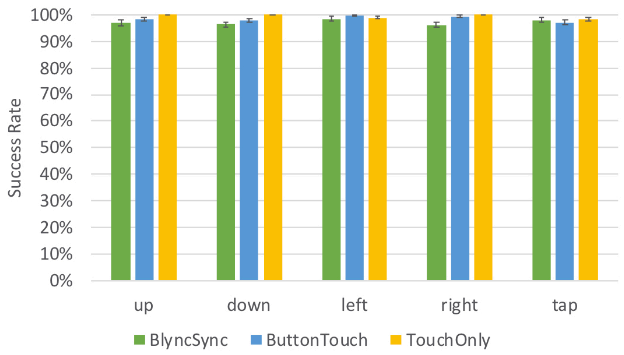

Figure 9. Success Rates for the techniques and gestures. Error bars represent standard error for Figures 9, 10, 11 and 12.

Design

A repeated measures within-participant design was used. The independent variables were *Technique* (*TouchOnly*, *BlyncSync*, *ButtonTouch*) and *Gesture* (*Up, Down, Left, Right, Tap)*. For each technique, participants were asked to perform six blocks of trials, each consisting of 20 trials (4 for each gesture, in random order). The ordering of technique was fully counter-balanced, with an optional 2-minute break between techniques. A short warmup was given to familiarize the participants with each of the techniques.

Results

Success Rates

A repeated measures analysis of variance indicated that neither *Technique* nor *Gesture* had a significant effect on success rates. The success rates were 97.1% for *BlyncSync*, 98.4% for *ButtonTouch,* and 99.4% for *TouchOnly* (Figure 9). This was an encouraging result, demonstrating that participants are able to perform the BlyncSync gestures with a high degree of success. For ButtonTouch, most failed trials (1.5%) were due to omitting the button press. The BlyncSync errors were all due to synchronization offsets, mostly for when the blink event occurred too late (2.7%), and the remaining due to the blink occurring too early (0.2%).

Completion Time

A repeated measures analysis of variance revealed a significant effect of *Technique* ($F_{2,\ 22} = 38.2$, $p < 0.001$). *TouchOnly* had the fastest average time (0.56s), followed by BlyncSync (0.71s), and ButtonTouch (1.06) (Figure 10). A post-hoc Tukey's test showed that the differences between each pair were significant ($p < .05$). There was no effect of *Gesture* on completion times [10]. It was encouraging to see BlyncSync was 33% faster than the baseline ButtonTouch, while adding only 150ms overhead cost to TouchOnly. The average blink duration was 120.3ms, 50.3ms longer than the natural blinks observed in study 1. This is consistent with prior work which has shown similar differences [9].

Offsets Analysis

For BlyncSync trials, we further analyzed the offsets between the blink and touch events. As shown in Figure 11, we found the average absolute offsets are well within the 250ms threshold we set for the study, indicating that participants were able to accurately synchronize their touch and blink events. The average offset was 78ms. However, offset values are lower for the start time (65ms for Offset$_s$ and 91ms for Offset$_e$). As such, it could be interesting to define separate thresholds for the start and end offset values. The relationship between the true positive rates and the offset threshold level is shown in Figure 12. When the offset is set to 250ms, a true positive rate of 97% is achieved and the standard error starts to converge (200ms: 5.15%, 250ms: 1.65%, 300ms: 1.21%, 350ms: 1.03%).

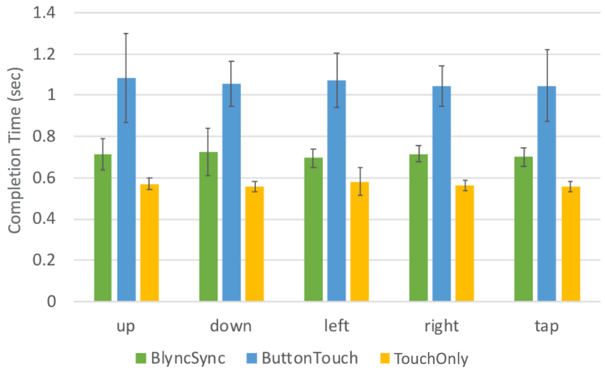

Figure 10. Completion times for the techniques and gestures.

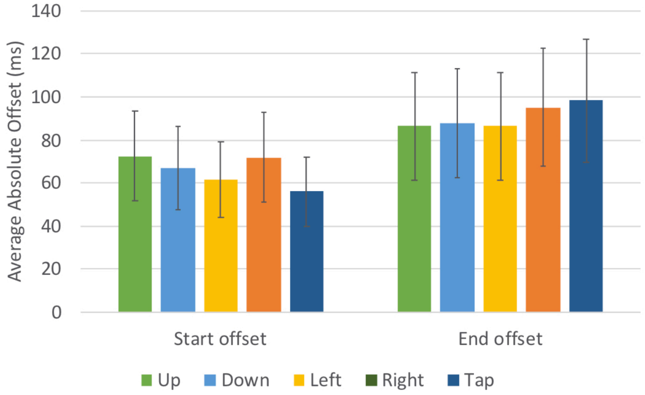

Figure 11. The offset values in the start and end times of synchronous touch and blink events.

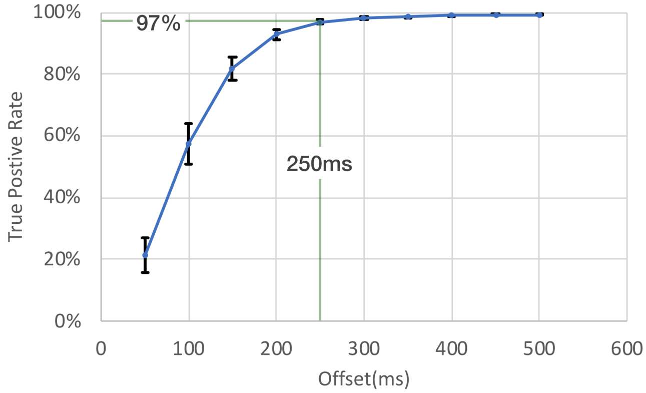

Figure 12. Relationship between offset and true positive rate. When the offset is set to 250ms, a true positive rate of 97% is achieved and the standard error starts to converge.

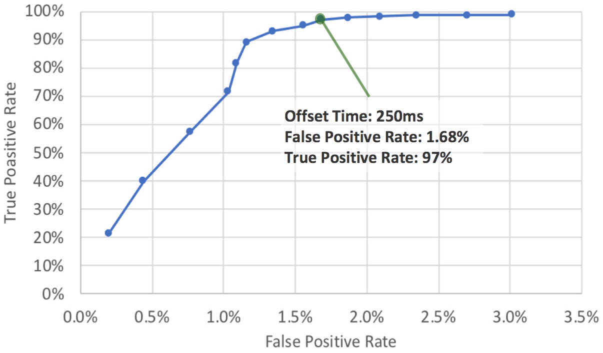

Figure 13. An ROC curve indicating the tradeoff between false positives and true positives.

Combining this data with the results from Study 1, we can plot the ROC curve showing the relationship between true positives and false positives at different offset values (Figure 13). Based on this data, we would suggest future implementations consider our chosen offset of 250ms which elicited a true positive rate of 97% while limiting the false positive rate to 1.68%.

Subjective Feedback and Observations

A number of participants commented that it was actually fun to use: e.g. '*the blinking part was fun*.' (P3). Once users got accustomed to the novel input paradigm, they were able to use the technique efficiently and with ease, as evident by our study results. The learnability of BlyncSync is sufficiently high as we observed every participant was able to consistently succeed in performing BlyncSync after no longer than 3 minutes of use. Moreover, we did not observe any participants struggling to use the technique, experience frustrations, and none reported eye fatigue.

In summary, our study showed that BlyncSync offers a new gesture delimiter to increase the input vocabulary for smartwatches, has minimal overhead (150ms) compared to standard touch, and outperforms a commonly used baseline modifier. In the next section, we discuss several applications of BlyncSync gestures.

SAMPLE MOCKUP APPLICATIONS

BlyncSync occupies a unique application space where blink and touch delimit each other. Therefore, we see two key classes of use for BlyncSync gestures: 1) Blink-augmented touch input to increase the input vocabulary of smartwatches and 2) Touch-augmented gaze input, to enable the use of gaze-based interactions, while avoiding false activations. In this section, we demonstrate both of these concepts by proposing two mock-up applications that utilize BlyncSync:

1. Smartwatch OS Shortcuts (*Blink-augmented touch*)
2. Gaze-Based IoT Control (*Touch-augmented gaze*)

Mockup 1: Smartwatch OS Shortcuts

System-level functions, such as calling the main menu or an app, are usually achieved by hardware buttons on smartwatches or mobile phones. This is necessary because gestures on the display are reserved for the app in use. However, it is not feasible or desirable to allocate too many buttons on the hardware, and as shown in Study 2, moving the finger to press a button adds time to the input transaction.

Therefore, we propose to use BlyncSync gestures as always-available gestures to perform OS Shortcuts, which can complement existing hardware buttons. We developed a proof-of-concept application on a Samsung Galaxy watch. The system was developed using JavaScript and Tizen Application APIs.

The two hardware buttons of the smartwatch already support exiting an application (without terminating it) and returning to the home menu. In addition, we mapped five frequently-used system functions to the five BlyncSync gestures. *BlyncSyncLeft* and *BlyncSyncRight* switch to the last/next application. *BlyncSyncDown* closes and terminates the current application and *BlyncSyncUp* shows all recently used applications. Last, *BlyncSyncTap* locks the screen, putting the device to sleep (Figure 14).

The use of these gestures would be more efficient than typical approaches on commercial smartwatches. For example, to switch between two apps, a user would need to repeatedly use a physical button to return to a home screen, and manually tap on the app of interest. We note that in practice, it would be possible to map any combination of smartwatch OS or application functions to the BlyncSync gestures, potentially configured by the end users.

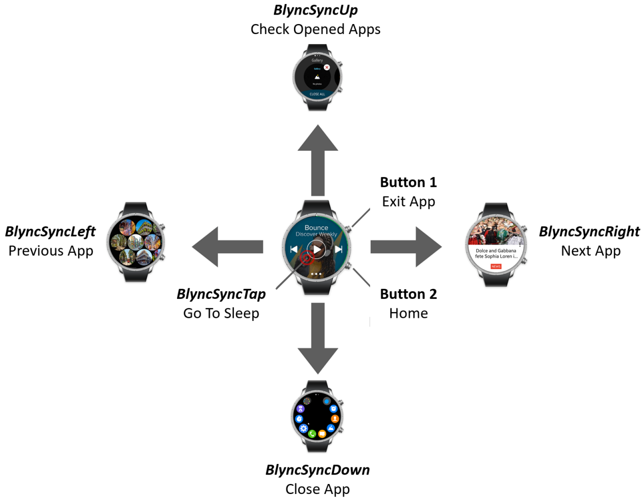

Figure 14. Illustration of using BlyncSync for Smartwatch OS Shortcuts. *BlyncSyncLeft* and *BlyncSyncRight* switch between apps. *BlyncSyncDown* closes and terminates the current app. *BlyncSyncUp* shows recently used apps. *BlyncSyncTap* locks the screen, putting the device to sleep

Mockup 2: Gaze-Based IoT Control

Using gaze to control appliances has been proposed by previous smart environment research [54, 55, 1]. Despite the rapidness of gaze control, it suffers the false activation problems we discussed previously. To address false positives, AmbiGaze [58] utilized smooth pursuits to trigger different smart home devices, but the process is time-consuming, lasting longer than 7 seconds to activate and trigger an object. We instead propose to use BlyncSync for smart environment control, by looking at an appliance and performing a BlyncSync gesture. We demonstrate this concept with three devices – a set of smart lightbulbs, a Bluetooth speaker, and a large display. In all examples, object selection is based on calibrated eye gaze direction. Future work could apply more advanced object recognition techniques [33, 51].

For the Large Display scenario, the user could display a notification which they received on their smartwatch. When the notification was received on the smartwatch, a *BlyncSyncUp* while looking at the large display would cause the details of the message to be shown on it, allowing the user to read the full details. This system was mocked up by connecting the laptop which detected the BlyncSync gesture to the display through HDMI. Python code was used to launch a webpage associated with the notification.

To interact with a smart light, the user would look at it, and then use a *BlyncSyncUp* to turn it on, a *BlyncSyncTap* to switch colors, and a *BlyncSyncDown* to turn it off (Figure 15a). The system was implemented with Philip Hue Light and Hue API and controlled through a Python web server.

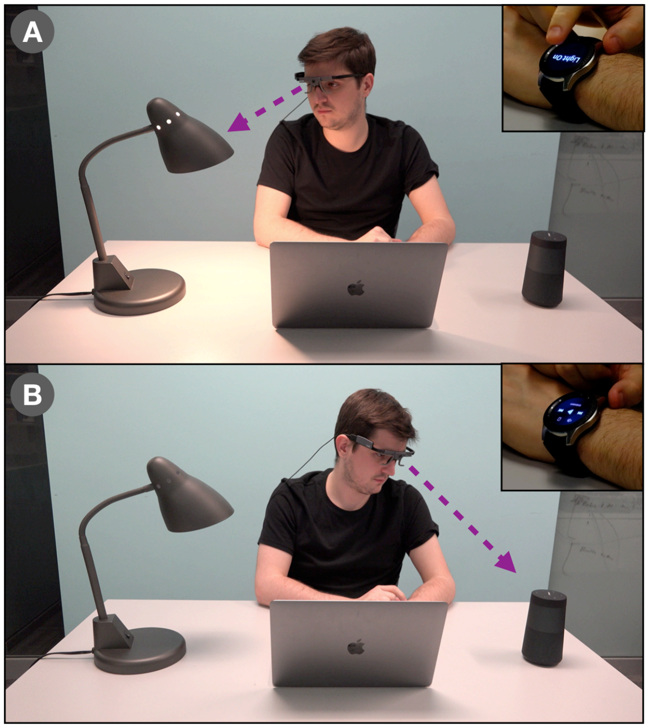

Figure 15. Gaze-based IoT Control. A) *BlyncSyncUp* while looking at a lamp turns it on. B) *BlyncSyncTap* while looking at a speaker retrieves a contextual UI on the watch.

For the Bluetooth speaker we explored the idea of retrieving a contextual user interface. If the user was looking at the speaker, a *BlyncSyncTap* would launch a music player app on the smartwatch (Figure 15b). The user could then control the playback of music on the speaker to interact with the smartwatch with normal touch, blending the use of both BlyncSync gestures and normal touch. This example was implemented using the two-way connection between the smartwatch and laptop server. When the server detected the *BlyncSyncTap,* it would send a message to the smartwatch to launch its native music app.

The advantage of using BlyncSync to control a smart environment is two-fold. First, BlyncSync solves the false activation problems of using gaze gestures, which enables instant activations and deactivations of smart devices. Second, complicated controls of devices can be transferred to the smartwatch, reducing the workload of eye input.

DISCUSSION AND FUTURE WORK

The results of our second study showed strong potential for the BlyncSync gestures which this paper introduces. Users were able to use the technique efficiently and accurately, with improved performance compared to a baseline mode switching technique. Our described sample applications demonstrate some of the opportunities that may be enabled by utilizing BlyncSync gestures in two main contexts (blink-augmented touch, and touch-augmented gaze).

We have also introduced a broader design space of touch & blink gestures, for which we have only started to explore. As such, there are many important questions and issues surrounding our work which warrant discussion.

Eye Tracking Technology

Eye tracking technology has been slowly improving over the years. Commercial products are now able to track the eyes precisely with light-weight wearable glasses. In particular, existing smart glasses able to capture blink events [22] show great potential for the use of BlyncSync in the near future. The eye tracker we used utilizes infrared cameras that detect the pupils with computer vision algorithms instead of measuring the eyelid movements, which might better characterize a blink event. Therefore, there is some ambiguity in detecting the timestamps of the blink start and end frames, i.e., when the pupils are not blocked by eyelids. Furthermore, optical blink detection is less consistent for users with glasses and contact lenses. Incorporating EEG or EOG information, e.g. JINS MEME [22], to capture the electrical signals of blinking could potentially be helpful. Furthermore, algorithms may need to be improved, and designed with power consumption in mind, if they are to be performed on the smartwatch itself. Alternatively, computation could be offloaded to a paired smartphone for greater performance.

Learnability and Complexity of Gestures

Since our goal was to find a set of gestures that would be robust to false positives and easy to perform, we only considered the most basic gestures from our described design space. It would still be interesting to design complex gestures when efficiency and simplicity is not a core priority. For example, one can use the sequential combination of BlyncSync and standard touch gestures for user authentication. In addition, other considerations in the design space could also be revisited, such as longer event durations or rhythm-based gestures [15, 41, 61].

One benefit of the simplicity of our technique was that participants were able to learn the technique quickly without the need for advanced guidance and training techniques. If more complex gestures were introduced, it would be interesting to explore techniques to improve the discoverability and learnability of the gestures, such as the use of visual guidance or feedforward [7].

In the Wild Evaluations and Implementations

Our studies were conducted in a controlled laboratory environment. Though beyond the scope of this paper, we plan to conduct an in-the-wild evaluation to observe whether users would behave differently in their daily lives. It may also be useful to validate our false positive findings from Study 1 in the context of real-world smartwatch usage. In particular, the touch activations elicited in this study are much higher than what would be observed in real-world usage [34, 49] and the blink rates might vary under extreme conditions, e.g. fatigue [56]. Furthermore, the sample applications we demonstrated were only partially implemented, enough to provide a sense of the interaction that we envision. Future work could develop robust versions of these applications, to enable deployments and in-depth user evaluations.

Avoid Conflicts with Common Touch

The recognition of BlyncSync gestures requires the system to wait for a potential blink after a touch event occurs. Therefore, the recognition of touch events needs to be slightly delayed. We note that this is a long-standing UI architecture challenge not unique to our gesture. For example, many multi-touch interfaces offer different behaviors based on the number of contact points (e.g. 1 touch for drag, 2 touch for stretch). Such systems similarly need to consider how to handle a first event prior to receiving a second event that may or may not ever occur. BlyncSync could address this challenge by performing the classification immediately upon the touch release, if the eye is open. In rare cases when the eye is closed upon touch release, a small delay (<150ms) could be injected to wait for the potential end of the BlyncSync gesture. We note, in such cases, the user's eyes would be closed, so this delay may not be perceived.

False Activations and Error Recovery

The results of Study 1 indicate that because start and end times must be synchronized, the number of non-intentional blinks while touching/gesturing is low. However, in rare instances where false activations do occur, some method of recovery should be provided. Future work could explore potential error recovery techniques specific to BlyncSync.

Generalizability to Other Platforms

Since BlyncSync provides a mutual delimiter for touch and eye input, it can be extended to enhance interaction on other touch platforms. For instance, it could be used in a similar manner on smartphones, tablets, and large touch-enabled display; It would also be interesting to see how the device form factor would influence the event rates observed in our first study. Moreover, it may be possible to use BlyncSync as a trigger in augmented reality platforms, as an alternative to in-air gestures, such as air tap. This could potentially reduce fatigue and increase gesture recognition accuracy. At a higher level, we believe the concept of *mutual delimiters* could be generalized to other mixed-modality platforms, similar to how *mutual disambiguation* has been explored across a range of modalities [43].

Contrasting to Existing Research

BlyncSync contributes a novel method for increasing the vocabulary of smartwatch input, while at the same time, offers a solution to the false activation problem of blink-based input. We acknowledge that BlyncSync might have on par or less improvement on input expressivity compared to other smartwatch input techniques (e.g. [41]), while requiring additional efforts of wearing an eye tracker. Nonetheless, it is encouraging to see our study showing BlyncSync is efficient and accurate relative to prior techniques which have used gaze, e.g. Orbits. The average completion time of a BlyncSync was 0.71s, whereas Orbits utilizes a recognition window size of 1s – meaning its performance time would be greater than that. Furthermore, Orbits reports a false activation rate of 2.1% and a true positive rate of 83%, whereas BlyncSync elicits a false activation rate of 1.68% and a true positive rate of 97%.

That being said, our technique was designed for different contexts of use, and we have not conducted a formal comparison to prior research systems under equivalent task conditions. Future work could perhaps survey the existing work, create a taxonomy of smartwatch input techniques and eye-based mobile interaction techniques, and contrast each of their benefits and limitations. Furthermore, since the studies in the paper focus on validating BlyncSync in smartwatch context, thoroughly investigating BlyncSync as a confirmation mechanism for gaze-based input would be valuable as well.

CONCLUSION

We have presented BlyncSync, a set of multi-modal smartwatch gestures that were designed as input delimiters for both touch and eye input. Our work introduces the concept of *mutual delimiters* and offers a design space of Touch & Blink gestures. Our experiments showed that the BlyncSync gestures are resistant to false positives, while still achieving high true positive rates. Since BlyncSync is easy to perform and unlikely to be accidentally invoked, it provides an always-active input method for smartwatch interaction. Based on the study results, we further provide insights to the trade-off of selecting activation thresholds, which could enable deployments of BlyncSync within commercial products. Furthermore, since the synchronous touch and eye events serve as mutual delimiters, BlyncSync spans a unique application space that covers both smartwatch interaction and gaze input, as demonstrated by our sample applications. We hope our technique will serve as important groundwork for future work on multimodal wearable input.

ACKNOWLEDGEMENTS

The authors thank Adrien Chaffangeon, Jiannan Li, and Zhuoyue Lyu for their participation in the paper video.

REFERENCES

[1] Mosa’ad Al-Abdulmunem and Stella T. Briggs. 1999. Spontaneous Blink Rate of a Normal Population Sample in International Contact Lens Clinic Volume 26, Issue 2, March–April 1999, Pages 29-32 DOI: https://doi.org/10.1016/S0892-8967(99)00016-4

[2] Hassoumi Almoctar, Pourang Irani, Vsevolod Peysakhovich, and Christophe Hurter. 2018. Path Word: A Multimodal Password Entry Method for Ad-hoc Authentication Based on Digits' Shape and Smooth Pursuit Eye Movements. In Proceedings of the 20th ACM International Conference on Multimodal Interaction (ICMI '18). DOI: https://doi.org/10.1145/3242969.3243008

[3] Ashbrook, D. L. (2010). *Enabling mobile microinteractions* (Doctoral dissertation, Georgia Institute of Technology).

[4] Daniel L. Ashbrook, James R. Clawson, Kent Lyons, Thad E. Starner, and Nirmal Patel. 2008. Quickdraw: the impact of mobility and on-body placement on device access time. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08). ACM, New York, NY, USA, 219-222. DOI: https://doi.org/10.1145/1357054.1357092

[5] Daniel Ashbrook, Kent Lyons, and Thad Starner. 2008. 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). ACM, New York, NY, USA, 311-314. DOI= http://dx.doi.org/10.1145/1409240.1409276

[6] Behrooz Ashtiani and I. Scott MacKenzie. 2010. BlinkWrite2: an improved text entry method using eye blinks. In Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications (ETRA '10). ACM, New York, NY, USA, 339-345. DOI= http://dx.doi.org/10.1145/1743666.1743742

[7] Olivier Bau and Wendy E. Mackay. 2008. OctoPocus: a dynamic guide for learning gesture-based command sets. In Proceedings of the 21st annual ACM symposium on User interface software and technology (UIST '08). ACM, New York, NY, USA, 37-46. DOI: https://doi.org/10.1145/1449715.1449724

[8] Benedetto, Simone & Pedrotti, Marco & Minin, Luca & Baccino, Thierry & Re, Alessandra & Montanari, Roberto. (2011). Driver workload and eye blink duration. Transportation Research Part F: Traffic Psychology and Behaviour. 14. 199-208. DOI: 10.1016/j.trf.2010.12.00 1

[9] Matteo Bologna, Rocco Agostino, Bruno Gregori, Daniele Belvisi, Donatella Ottaviani, Carlo Colosimo, Giovanni Fabbrini, Alfredo Berardelli, Voluntary, spontaneous and reflex blinking in patients with clinically probable progressive supranuclear palsy, Brain, Volume 132, Issue 2, February 2009, Pages 502–510, https://doi.org/10.1093/brain/awn317

[10] Xiang Cao and Shumin Zhai. 2007. Modeling human performance of pen stroke gestures. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07). ACM, New York, NY, USA, 1495-1504. DOI: https://doi.org/10.1145/1240624.1240850

[11] Xiang 'Anthony' Chen, Tovi Grossman, and George Fitzmaurice. 2014. Swipeboard: a text entry technique for ultra-small interfaces that supports novice to expert transitions. In Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST '14). ACM, New York, NY, USA, 615-620. DOI: https://doi.org/10.1145/2642918.2647354

[12] Flaviu Cristian. 1989. Probabilistic clock synchronization. In Distributed Computing. https://doi.org/10.1007/BF01784024

[13] Harishchandra Dubey, Jon C. Goldberg, Mohammadreza Abtahi, Leslie Mahler, and Kunal Mankodiya. 2015. EchoWear: smartwatch technology for voice and speech treatments of patients with Parkinson's disease. In Proceedings of the conference on Wireless Health (WH '15). ACM, New York, NY, USA, Article 15, 8 pages. DOI: https://doi.org/10.1145/2811780.2811957

[14] Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2015. Orbits: Gaze Interaction for Smart Watches using Smooth Pursuit Eye Movements. In Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15). ACM, New York, NY, USA, 457-466. DOI: https://doi.org/10.1145/2807442.2807499

[15] Emilien Ghomi, Guillaume Faure, Stéphane Huot, Olivier Chapuis, and Michel Beaudouin-Lafon. 2012. Using rhythmic patterns as an input method. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12). ACM, New York, NY, USA, 1253-1262. DOI: https://doi.org/10.1145/2207676.2208579

[16] Chris Harrison and Scott E. Hudson. 2009. Abracadabra: wireless, high-precision, and unpowered finger input for very small mobile devices. In Proceedings of the 22nd annual ACM symposium on User interface software and technology (UIST '09). ACM, New York, NY, USA, 121-124. DOI: https://doi.org/10.1145/1622176.1622199

[17] Ken Hinckley. 2003. Synchronous gestures for multiple persons and computers. In Proceedings of the 16th annual ACM symposium on User interface software and technology (UIST ’03). Association for Computing Machinery, New York, NY, USA, 149–158. DOI:https://doi.org/10.1145/964696.964713

[18] Ken Hinckley, Patrick Baudisch, and Gonzalo Ramos, Francois Guimbretiere. 2005. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '05). ACM, New York, NY, USA, 451-460. DOI: https://doi.org/10.1145/1054972.1055035

[19] Holmquist L.E., Mattern F., Schiele B., Alahuhta P., Beigl5 M., Gellersen HW. (2001) Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts. In: Abowd G.D., Brumitt B., Shafer S. (eds) Ubicomp 2001: Ubiquitous Computing. UbiComp 2001. Lecture Notes in Computer Science, vol 2201. Springer, Berlin, Heidelberg

[20] Christian Holz and Patrick Baudisch. 2010. The generalized perceived input point model and how to double touch accuracy by extracting fingerprints. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10). ACM, New York, NY, USA, 581-590. DOI: https://doi.org/10.1145/1753326.1753413

[21] Da-Yuan Huang, Ming-Chang Tsai, Ying-Chao Tung, Min-Lun Tsai, Yen-Ting Yeh, Liwei Chan, Yi-Ping Hung, and Mike Y. Chen. 2014. TouchSense: expanding touchscreen input vocabulary using different areas of users' finger pads. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14). ACM, New York, NY, USA, 189-192. DOI: https://doi.org/10.1145/2556288.2557258

[22] Shoya Ishimaru, Kai Kunze, Katsuma Tanaka, Yuji Uema, Koichi Kise, and Masahiko Inami. 2015. Smart Eyewear for Interaction and Activity Recognition. In Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15). ACM, New York, NY, USA, 307-310. DOI: https://doi.org/10.1145/2702613.2725449

[23] Robert J. K. Jacob. 1990. What you look at is what you get: eye movement-based interaction techniques. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90), Jane Carrasco Chew and John Whiteside (Eds.). ACM, New York, NY, USA, 11-18. DOI= http://dx.doi.org/10.1145/97243.97246

[24] Mohamed Khamis, Florian Alt, Mariam Hassib, Emanuel von Zezschwitz, Regina Hasholzner, and Andreas Bulling. 2016. GazeTouchPass: Multimodal Authentication Using Gaze and Touch on Mobile Devices. In Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16). ACM, New York, NY, USA, 2156-2164. DOI: https://doi.org/10.1145/2851581.2892314

[25] Mohamed Khamis, Mariam Hassib, Emanuel von Zezschwitz, Andreas Bulling, and Florian Alt. 2017. GazeTouchPIN: protecting sensitive data on mobile devices using secure multimodal authentication. In Proceedings of the 19th ACM International Conference on Multimodal Interaction (ICMI '17). ACM, New York, NY, USA, 446-450. DOI: https://doi.org/10.1145/3136755.3136809

[26] Jungsoo Kim, Jiasheng He, Kent Lyons, and Thad Starner. 2007. The Gesture Watch: A Wireless Contact free Gesture based Wrist Interface. In Proceedings of the 2007 11th IEEE International Symposium on Wearable Computers (ISWC '07). IEEE Computer Society, Washington, DC, USA, 1-8. DOI: 10.1109/ISWC.2007.4373770

[27] Kowalczyk, P. & Sawicki, D. Blink and wink detection as a control tool in multimodal interaction Multimedia Tools Appl (2019) 78: 13749. https://doi.org/10.1007/s11042-018-6554-8

[28] Sven Kratz and Michael Rohs. 2009. Hoverflow: exploring around-device interaction with IR distance sensors. In Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '09). ACM, New York, NY, USA, , Article 42 , 4 pages. DOI= http://dx.doi.org/10.1145/1613858.1613912

[29] Królak, A., & Strumiłło, P. (2012). Eye-blink detection system for human–computer interaction. Universal Access in the Information Society, 11(4), 409-419. https://doi.org/10.1007/s10209-011-0256-6

[30] Gordon Kurtenbach and William Buxton. 1991. Issues in combining marking and direct manipulation techniques. In Proceedings of the 4th annual ACM symposium on User interface software and technology (UIST '91). ACM, New York, NY, USA, 137-144. DOI= http://dx.doi.org/10.1145/120782.120797

[31] 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 Proceedings of the 27th annual ACM symposium on User interface software and technology(UIST '14) ACM, New York, NY, USA, 389-394. DOI: https://doi.org/10.1145/2642918.2647356

[32] Juyoung Lee, Shaurye Aggarwal, Jason Wu, Thad Starner, and Woontack Woo. 2019. SelfSync: exploring self-synchronous body-based hotword gestures for initiating interaction. In Proceedings of the 23rd International Symposium on Wearable Computers (ISWC ’19). Association for Computing Machinery, New York, NY, USA, 123–128. DOI:https://doi.org/10.1145/3341163.3347745

[33] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. 2015. SSD: Single Shot MultiBox Detector. CoRR abs/1512.02325 (2015).

[34] Xing Liu, Tianyu Chen, Feng Qian, Zhixiu Guo, Felix Xiaozhu Lin, Xiaofeng Wang, and Kai Chen. 2017. Characterizing Smartwatch Usage in the Wild. In Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '17). ACM, New York, NY, USA, 385-398. DOI: https://doi.org/10.1145/3081333.308135 1

[35] Ho-Man Colman Leung, Chi-Wing Fu, and Pheng-Ann Heng. 2018. TwistIn: Tangible Authentication of Smart Devices via Motion Co-analysis with a Smartwatch. Proc. ACM Interact. Mob. Wearable Ubiquitous

Technol. 2, 2, Article 72 (July 2018), 24 pages. DOI: https://doi.org/10.1145/3214275

[36] Kent Lyons, David Nguyen, Daniel Ashbrook, and Sean White. 2012. Facet: a multi-segment wrist worn system. In Proceedings of the 25th annual ACM symposium on User interface software and technology (UIST '12). ACM, New York, NY, USA, 123-130 ACM, New York, NY, USA, 123-130. DOI: https://doi.org/10.1145/2380116.2380134

[37] I. Scott MacKenzie and Behrooz Ashtiani. 2011. BlinkWrite: efficient text entry using eye blinks. Univers. Access Inf. Soc. 10, 1 (March 2011), 69-80. DOI: 10.1007/s10209-010-0188-6

[38] Eric Missimer and Margrit Betke. 2010. Blink and wink detection for mouse pointer control. In Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments (PETRA '10), Fillia Makedon, Ilias Maglogiannis, and Sarantos Kapidakis (Eds.) ACM, New York, NY, USA, , Article 23 , 8 pages. DOI= http://dx.doi.org/10.1145/1839294.1839322

[39] Mukherjee, K., & Chatterjee, D. (2015, January). Augmentative and alternative communication device based on eye-blink detection and conversion to Morse-code to aid paralyzed individuals. In 2015 International Conference on Communication, Information & Computing Technology (ICCICT) (pp. 1-5). IEEE. DOI:10.1109/iccict.2015.7045754

[40] Ian Oakley and Doyoung Lee. 2014. Interaction on the edge: offset sensing for small devices. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14). ACM, New York, NY, USA, 169-178. DOI: https://doi.org/10.1145/2556288.2557138

[41] Ian Oakley, DoYoung Lee, MD. Rasel Islam, and Augusto Esteves. 2015. Beats: Tapping Gestures for Smart Watches. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15). ACM, New York, NY, USA, 1237-1246. DOI: https://doi.org/10.1145/2702123.2702226

[42] Stephen Oney, Chris Harrison, Amy Ogan, and Jason Wiese. 2013. ZoomBoard: a diminutive qwerty soft keyboard using iterative zooming for ultra-small devices. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13) ACM, New York, NY, USA, 2799-2802. DOI: https://doi.org/10.1145/2470654.2481387

[43] Sharon Oviatt. 1999. Mutual disambiguation of recognition errors in a multimodal architecture. In Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI '99). ACM, New York, NY, USA, 576-583. DOI= http://dx.doi.org/10.1145/302979.303163

[44] Lyelle L. Palmer. Inability to Wink an Eye and Eye Dominance. 1976. In Perceptual and Motor Skills. DOI: 10.2466/pms.1976.42.3.825

[45] Jerome Pasquero, Scott J. Stobbe, and Noel Stonehouse. 2011. A haptic wristwatch for eyes-free interactions. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11). ACM, New York, NY, USA, 3257-3266. DOI: https://doi.org/10.1145/1978942.1979425

[46] Simon T. Perrault, Eric Lecolinet, James Eagan, and Yves Guiard. 2013. 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). ACM, New York, NY, USA, 1451-1460. DOI: https://doi.org/10.1145/2470654.2466192

[47] Ken Pfeuffer, Jason Alexander, Ming Ki Chong, and Hans Gellersen. 2014. Gaze-touch: combining gaze with multi-touch for interaction on the same surface. In Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST '14). ACM, New York, NY, USA, 509-518. DOI: https://doi.org/10.1145/2642918.2647397

[48] Ken Pfeuffer, Jason Alexander, Ming Ki Chong, Yanxia Zhang, and Hans Gellersen. 2015. Gaze-Shifting: Direct-Indirect Input with Pen and Touch Modulated by Gaze. In Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST ’15). Association for Computing Machinery, New York, NY, USA, 373–383. DOI: https://doi.org/10.1145/2807442.2807460

[49] Stefania Pizza, Barry Brown, Donald McMillan, and Airi Lampinen. 2016. Smartwatch in *vivo*. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). ACM, New York, NY, USA, 5456-5469. DOI: https://doi.org/10.1145/2858036.2858522

[50] Gonzalo Ramos, Kenneth Hinckley, Andy Wilson & Raman Sarin (2009) Synchronous Gestures in Multi-Display Environments, Human–Computer Interaction, 24:1-2, 117-169, DOI: 10.1080/07370020902739288

[51] Joseph Redmon and Ali Farhadi. 2016. YOLO9000: Better, Faster, Stronger. CoRR abs/1612.08242 (2016). http://arxiv.org/abs/1612.08242

[52] Rekimoto J., Ayatsuka Y., Kohno M. (2003) SyncTap: An Interaction Technique for Mobile Networking. In: Chittaro L. (eds) Human-Computer Interaction with Mobile Devices and Services. Mobile HCI 2003. Lecture Notes in Computer Science, vol 2795. Springer, Berlin, Heidelberg

[53] Gabriel Reyes, Jason Wu, Nikita Juneja, Maxim Goldshtein, W. Keith Edwards, Gregory D. Abowd, and Thad Starner. 2018. SynchroWatch: One-Handed Synchronous Smartwatch Gestures Using Correlation and Magnetic Sensing. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 1, 4, Article 158 (Jan. 2018), 26 pages. https://doi.org/10.1145/3161162

[54] Jeffrey S. Shell, Roel Vertegaal, and Alexander W. Skaburskis. 2003. EyePliances: attention-seeking devices that respond to visual attention. In CHI '03

Extended Abstracts on Human Factors in Computing Systems (CHI EA '03). ACM, New York, NY, USA, 770-771. DOI= http://dx.doi.org/10.1145/765891.765981

[55] Jeffrey S. Shell, Roel Vertegaal, Daniel Cheng, Alexander W. Skaburskis, Changuk Sohn, A. James Stewart, Omar Aoudeh, and Connor Dickie. 2004. ECSGlasses and EyePliances: using attention to open sociable windows of interaction. In Proceedings of the 2004 symposium on Eye tracking research & applications (ETRA '04). ACM, New York, NY, USA, 93-100. DOI: https://doi.org/10.1145/968363.968384

[56] Stern, J. A., Boyer, D., & Schroeder, D. (1994). Blink Rate: A Possible Measure of Fatigue. Human Factors, 36(2), 285–297. https://doi.org/10.1177/001872089403600209

[57] Eduardo Velloso, Marcus Carter, Joshua Newn, Augusto Esteves, Christopher Clarke, and Hans Gellersen. 2017. Motion Correlation: Selecting Objects by Matching Their Movement. ACM Trans. Comput.-Hum. Interact. 24, 3, Article 22 (April 2017), 35 pages. DOI:https://doi.org/10.1145/3064937

[58] Eduardo Velloso, Markus Wirth, Christian Weichel, Augusto Esteves, and Hans Gellersen. 2016. AmbiGaze: Direct Control of Ambient Devices by Gaze. In Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16). ACM, New York, NY, USA, 812-817. DOI: https://doi.org/10.1145/2901790.2901867

[59] Roel Vertegaal, Aadil Mamuji, Changuk Sohn, and Daniel Cheng. 2005. Media eyepliances: using eye tracking for remote control focus selection of appliances. In CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05). ACM, New York, NY, USA, 1861-1864. DOI= http://dx.doi.org/10.1145/1056808.1057041

[60] Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: spontaneous interaction with displays based on smooth pursuit eye movement and moving targets. In Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing (UbiComp '13). ACM, New York, NY, USA, 439-448. DOI: https://doi.org/10.1145/2493432.2493477

[61] Jacob Otto Wobbrock. 2009. TapSongs: tapping rhythm-based passwords on a single binary sensor. In Proceedings of the 22nd annual ACM symposium on User interface software and technology (UIST '09). ACM, New York, NY, USA, 93-96. DOI: https://doi.org/10.1145/1622176.1622194

[62] Jason Wu, Cooper Colglazier, Adhithya Ravishankar, Yuyan Duan, Yuanbo Wang, Thomas Ploetz, and Thad Starner. 2018. Seesaw: Rapid One-Handed Synchronous Gesture Interface for Smartwatches. In Proceedings of the 2018 ACM International Symposium on Wearable Computers (ISWC '18). ACM, New York, NY, USA, 17–20. https://doi.org/10.1145/3267242.3267251

[63] Haijun Xia, Tovi Grossman, and George Fitzmaurice. 2015. 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). ACM, New York, NY, USA, 447-456. DOI: https://doi.org/10.1145/2807442.2807500

[64] Robert Xiao, Gierad Laput, and Chris Harrison. 2014. Expanding the input expressivity of smartwatches with mechanical pan, twist, tilt and click. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14). ACM, New York, NY, USA, 193-196. DOI: https://doi.org/10.1145/2556288.2557017

[65] York M, Ong J, Robbins JC: Variation in blink rate associated with contact lens wear and task difficulty. Am J Optom Am Acad Optom 1971;48:461–466.

[66] Shumin Zhai, Carlos Morimoto, and Steven Ihde. 1999. Manual and gaze input cascaded (MAGIC) pointing. In Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI '99). ACM, New York, NY, USA, 246-253. DOI=http://dx.doi.org/10.1145/302979.303053

[67] Cheng Zhang, Xiaoxuan Wang, Anandghan Waghmare, Sumeet Jain, Thomas Ploetz, Omer T. Inan, Thad E. Starner, and Gregory D. Abowd. 2017. FingOrbits: Interaction with Wearables Using Synchronized Thumb Movements. In Proceedings of the 2017 ACM International Symposium on Wearable Computers (ISWC '17). ACM, New York, NY, USA, 62–65. https://doi.org/10.1145/3123021.3123041

[68] Qiaohui Zhang, Atsumi Imamiya, Kentaro Go, Xiaoyang Mao. A Gaze and Speech Multimodal Interface. 2004. In Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCSW'04) DOI: 10.1109/ICDCSW.2004.1284033

[69] Junhan Zhou, Yang Zhang, Gierad Laput, and Chris Harrison. 2016. AuraSense: Enabling Expressive Around-Smartwatch Interactions with Electric Field Sensing. In Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16). ACM, New York, NY, USA, 81-86. DOI: https://doi.org/10.1145/2984511.2984568