SynchroWatch: One-Handed Synchronous Smartwatch Gestures Using Correlation and Magnetic Sensing

GABRIEL REYES, School of Interactive Computing, Georgia Institute of Technology, USA
JASON WU, School of Interactive Computing, Georgia Institute of Technology, USA
NIKITA JUNEJA, School of Interactive Computing, Georgia Institute of Technology, USA
MAXIM GOLDSHTEIN, School of Aerospace Engineering, Georgia Institute of Technology, USA
W. KEITH EDWARDS, School of Interactive Computing, Georgia Institute of Technology, USA
GREGORY D. ABOWD, School of Interactive Computing, Georgia Institute of Technology, USA
THAD STARNER, School of Interactive Computing, Georgia Institute of Technology, USA

SynchroWatch is a one-handed interaction technique for smartwatches that uses rhythmic correlation between a user's thumb movement and on-screen blinking controls. Our technique uses magnetic sensing to track the synchronous extension and reposition of the thumb, augmented with a passive magnetic ring. The system measures the relative changes in the magnetic field induced by the required thumb movement and uses a time-shifted correlation approach with a reference waveform for detection of synchrony. We evaluated the technique during three distraction tasks with varying degrees of hand and finger movement: active walking, browsing on a computer, and relaxing while watching online videos. Our initial offline results suggest that intentional synchronous gestures can be distinguished from other movement. A second evaluation using a live implementation of the system running on a smartwatch suggests that this technique is viable for gestures used to respond to notifications or issue commands. Finally, we present three demonstration applications that highlight the technique running in real-time on the smartwatch.

CCS Concepts: • **Human-centered computing → Gestural input**;

Additional Key Words and Phrases: wearable computing, interaction techniques, synchronous gestures, smartwatches, magnetic sensing, one-handed input

ACM Reference Format:
Gabriel Reyes, Jason Wu, Nikita Juneja, Maxim Goldshtein, W. Keith Edwards, Gregory D. Abowd, and Thad Starner. 2017. SynchroWatch: One-Handed Synchronous Smartwatch Gestures Using Correlation and Magnetic Sensing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 158 (December 2017), 26 pages. https://doi.org/10.1145/3161162

1 INTRODUCTION

The emergence of smartwatches has enabled new ways to glance at content on the wrist and provide input on-the-go using microinteractions [3]. The touchscreen and hardware buttons are the primary forms of input on the smartwatch today. While touch offers an intuitive modality, it is not suitable for many hands-busy situations as it also encumbers the hand not wearing the watch. Motion gestures are an alternate input modality that have

Gabriel Reyes, greyes@gatech.edu, School of Interactive Computing, Georgia Institute of Technology, 85 5th St NW, Atlanta, Georgia, USA.

2474-9567/2017/12-ART158 $15.00
https://doi.org/10.1145/3161162

Fig. 1. (A) Synchro is a one-handed interaction technique for smartwatches that uses rhythmic correlation between the user's thumb and binary blinking controls. (B) A demonstration application with a blinking dismiss to voicemail icon (in blue). (C) Blinking targets can be placed in a vertical layout, such as in this reading application. (D) A music player with four possible targets. A wrist flick gesture is used to toggle between pairs of binary targets. (E) The intent to 'sync' is detected when the thumb is repositioned (close to the hand) and icon color changes indicate detection.

been explored in research [2, 3, 24, 33] and adopted in commercial wearable devices for scrolling and selection[1]. Motion gestures may suffer from false positive activation during everyday use and there is a learning curve for users to remember what gestures correspond to a given action.

We propose SynchroWatch, an interaction technique that enables the user to perform one-handed binary *object selection* on the smartwatch using simple thumb motions (i.e., input) in response to on-screen stimuli (i.e., output). Our interface presents on-screen stimuli in the form of binary objects for selection which are blinking out of phase at the same frequency. As input, the user extends (away from the hand) and repositions (toward the hand) their thumb instrumented with a magnetic thumb ring to match the blinking frequency of the desired target icon (see Figure 1). Our interface is an example of "moving controls" and part of a larger body of work in synchronous gestures. The idea behind synchronous gestures is to represent objects available for selection by animating them (e.g., via motion, blinking, resizing, etc.) in the interface, have users identify a target by mimicking (e.g., via gaze tracking, thumb motion, hand movement, etc.) the specific stimulus, and use the correlation between the system's output and the user's input to determine if a selection is occurring and what object is selected.

We motivate the use of our synchronous interface through an example with a common scenario on the smartwatch — controlling a music player. The on-screen interface presents the user with multiple targets to select (e.g., next song, previous song, volume up, volume down). If the user wanted to select one of multiple targets on this interface, there are a number of ways to do so. One way is to assign a unique motion gesture of the wrist per target. However, that would be inefficient and require the user to remember which gesture to use for each target. The user could use a few directional gestures (e.g., swipe left and swipe right) to navigate the interface and another gesture to select a target. The same operations could be performed with the non-watch hand using hardware buttons or direct touch, which are the most common way of interacting today. In each of these cases, the user is expected to adapt to the interface or use the non-watch hand. Our approach using synchronous gestures instead modifies the interface to enable the quick selection of multiple targets with a simple rhythmic motion of the thumb on the watch hand.

[1]https://developer.android.com/wear/preview/features/gestures.html

Our technique leverages magnetic sensing and we use the smartwatch magnetometer to sense the continuous extension and reposition of the user's thumb, instrumented with a passive magnetic ring. The thumb movement approaches a sinusoidal waveform that is correlated to the blinking frequency of the target elements. The system supports binary selection by treating the turning points of the thumb extension and reposition as binary states (i.e., crest and trough of a sine wave) that correspond to the on-off states of a blinking target. There are a number of advantages to this approach: does not require users to memorize gestures, requires a small range of motion to interact, is intuitive, and has high discoverability.

SynchroWatch seeks to contribute a new technique for one-handed smartwatch input using *rhythmic correlation*, in particular matching a new projection feature derived from the watch's magnetometer signal with a known sinusoidal reference signal based on a simple harmonic motion model. Our correlation approach 1) eliminates the need for calibration or training, 2) minimizes false positive activation, and 3) allows operation while on-the-go. Synchronizing the known thumb movement to an object blinking at a known frequency/phase provides the user with guidance on how and when to perform input, and also facilitates the goal of eliminating the need for user training. The algorithm performs cross-correlation using the known blink frequency of targets and the input signal to compensate for any latency in the user's stimulus response. Synchrony also supports the goal of minimizing false positive activation by reducing the search space of matching sensor data. Finally, we seek to empower the user to perform smartwatch input while they're engaging in daily activities, such as walking, leveraging movement correlation and expected movement patterns.

Our work demonstrates the algorithm is robust and functions while in motion. We evaluated the system through two user studies. The first study is offline using sensor data collected with participants and analyzed post hoc. Based on the findings from the first evaluation, we implemented a prototype that runs in real-time on a smartwatch for a second study. Since the technique depends on relative and synchronous changes in the sensor data, it can be deployed on commodity watch platforms that are equipped with a magnetometer without knowing the exact location of the sensor inside the device. The second study runs in real-time on the device and provides users with live feedback of their performance by changing the color of the intended object based on the correlation score.

SynchroWatch gestures can be used to select between binary targets in various user scenarios. More specifically, when a notification (visual, acoustic, or haptic) arrives on the watch, the gesture can be used as a *response*. For example, the user can quickly dismiss a phone call to voicemail by syncing their thumb movement to a blinking dismiss icon. SynchroWatch can also be used as a *command* gesture, within the context of an application, to select between binary targets. It is possible to extend the expressivity of the gesture using a complementary input modality, such as a wrist flick gesture. A flick gesture can toggle between different pairs of blinking controls, expanding selection beyond the basic case of two targets. Finally, with additional work to reduce false positives to an extremely low level and have high true positive detection, SynchroWatch could be used as an *activation* gesture [36] if matched to a blinking LED around the bezel of the device. For example, the synchronized gesture could turn on the smartwatch display, activate another set of command gestures, or quickly launch an application bypassing the home screen. However, the focus of our current system is for notification response and command gesture scenarios.

Our work makes the following contributions:

- We describe an interaction technique for one-handed synchronous gestures on smartwatch devices that relies on synchrony between user movement and blinking on-screen controls.
- We describe a technique utilizing magnetic sensing with a passive thumb ring enabling one-handed, calibration-free, and user-independent gestures.
- We assess the technique using time-shifted correlation with a reference signal based on a simple harmonic motion model.

- We provide empirical evidence of the algorithmic performance through two user studies with eight participants in various distraction tasks (sitting, browsing on laptop, and walking), and compare performance versus *touch swiping*.

2 RELATED WORK

We position our contributions and discuss related work in the areas of smartwatch interaction techniques, approaches using magnetic sensing for interaction, the use of rhythm as input, and the space of synchronous gestures.

2.1 Smartwatch Interaction Techniques

The most common way of interacting with a smartwatch device is using the non-watch hand for touchscreen input or button controls. Speech recognition often is a secondary modality that enables high bandwidth dictation. While a number of projects have focused on expanding the input capabilities on smartwatches for two-handed input using around-device sensing and custom form factors [19, 32, 43–45], our work focuses on one-handed (the hand wearing the watch) discrete interactions on a commodity smartwatch.

In particular, one-handed interactions support situations in which the user's other hand may not be available if the person is carrying something or suffers from a disability. A few techniques have motivated the use of commodity sensing on a smartwatch and inspired the design of SynchroWatch in this paper. Serendipity [42] uses motion sensors in a smartwatch to distinguish fine-motor gestures such as pinching, rubbing, and tapping fingers. ViBand [26] presented an approach for one-handed input and other interactions using high-fidelity bio-acoustic sensing with the accelerometer and a custom kernel on the smartwatch. A tilting metaphor is another approach that has been used for menu navigation, selection, and text entry [17, 18]. Reyes et al. used the microphone in the smartwatch to detect non-voice acoustics with the mouth as an actuator for no-handed input [35].

Prior work also focuses on developing custom watch devices. Facet provided a multi-display wristband consisting of multiple independent smartwatch screens, enabling a rich set of touch input techniques [29]. Xiao et al. provided a multi-degree-of-freedom mechanical watch face that supports continuous 2D pan and twist, as well as binary tilt and click [44]. Laput et al. used proximity sensing [25] to enable "skin buttons" that can be activated by touching the skin around the watch. Oakley et al. use proximity sensing around the watch face to capture interaction on the edge of small devices [32]. WatchIt introduces a custom watch band for eyes-free interaction [34]. WristWhirl [16] presented a smartwatch band using proximity sensing to convert the wrist into an always-available joystick (i.e., to control a cursor) and piezo sensing to detect thumb pinches (i.e., for selection).

2.2 Magnetic Sensing for Interaction

Magnetic sensing has been explored extensively for mobile and wearable interactions [22, 27, 28]. While modifying the case or band around a smartwatch with electronics may provide additional interaction capabilities, it will also place varying degrees of power constraints on a device with limited battery capacity. Magnetic sensing can enable a battery-less approach to interaction and enable input around the smartwatch.

A number of projects have utilized magnetic rings for interaction. Ketabdar et al. [23] use the magnetometer in a mobile phone for around device interaction. A permanent magnet held or worn as a ring on the non-watch hand is used for interaction. The technique enables coarse gestures in 3D space around the device. The temporal pattern of the gesture affects the magnetic field around the mobile phone and features are used to determine which gesture is being performed. Abracadabra [19] is a magnetically driven input technique that offers users high fidelity finger input for smartwatches. A magnet is worn on the index finger of the non-watch hand and the technique extends the input area around the device for continuous cursor control and selection. Nenya [1] is a

magnetic ring worn as a wedding band which supports selection by twisting and selection by sliding it along the finger. The downside of all of these techniques is that they require the use of the non-watch hand. While possible, one-handed sliding with the Nenya was found to be uncomfortable and reduced performance. Finally, perhaps most similar to SynchroWatch, uTrack [9] is a one-handed input technique that uses a thumb magnet and two magnetometers on the ring finger to convert the movement of the thumb into a 3D continuous pointing device.

SynchroWatch does not require any additional hardware beyond the smartwatch and a small passive magnetic ring, eliminating the additional expense, bulkiness, or power requirements of other sensing approaches. Instead, SynchroWatch depends on periodic thumb motions matched to output stimuli on the watch display, and the movement of the magnetic thumb ring is sensed by the smartwatch's magnetometer. The interaction is designed to be robust to global motion (e.g., walking) enabling quick input on-the-go; the majority of other synchronous interfaces have only focused on stationary scenarios.

2.3 Rhythmic Patterns for Interaction

Interaction techniques that use the temporal dimension have been explored for a long time. Hinckley et al. [20] proposed an interaction metaphor for distributed sensing systems, highlighting single events co-occurring in time across multiple devices or people. For example, bumping two tablets together or two people wearing an accelerometer-augmented watch to sense shaking hands. In the tablet bumping scenario, each tablet experiences a roughly equal but opposite pattern of forces. Bumping the left side of a tablet versus striking the right side results in a similar sensor pattern, but with spikes in the opposite direction. This approach allows the system software to determine which side of each tablet has made contact.

Hudson et al. [21] presented an interaction technique that relies on detecting multiple 'whacking' events over time. Different from Hinckley et al.'s work, this approach looks at repetition over time with a pattern that is previously known to the user (e.g., whack, whack, whack) on a single mobile device, as opposed to across multiple devices.

More advanced uses of rhythmic patterns have begun to receive attention from the human-computer interaction communities [15]. Rhythmic interfaces are based on a regular pattern of recurring beats or related elements over time. Rhythmic interfaces present a number of advantages: they can be performed in a variety of situations, they can be coupled with output stimuli or feedback, and rhythm can be performed with different body parts and captured with different sensors.

Mobile implementations using rhythmic interaction have been used for filtering through a music library [6, 12]. Users could tap the screen or shake the device at a given tempo to trigger playing a song in their library. The authors introduced variability in the beat frequency to display ambiguity and allow users to adjust their actions based on the given on-screen feedback.

Other work focuses on coupling of input and output by periodic motion. In Resonant Bits [5], the coupling is explored in terms of resonance and how a system's continuous feedback can guide the user's rhythmic movement of the mobile phone. Over time, the movement patterns of the accelerometer accumulate to produce a visible excitation outcome for input control. In CycloStar [30], continuous closed loop movements modulating a sustained oscillation are used to support panning and zooming in touch interfaces.

2.4 Synchronous Gestures

Synchronous gesture interfaces are similar to rhythmic interaction systems as they expect the user to provide a continuous pattern of input over time. However, the key difference is that synchronous gesture interfaces provide the user with stimuli on when, and some cases how, to perform user input and correlate that input to the given stimuli.

Motion correlation is the most common form of synchronous gestures. Velloso et al. present an overview of motion correlation systems and guidelines for the design of correlation-based algorithms [38]. The idea behind motion correlation is to "represent available objects for selection by motion in the interface, have users identify a target by mimicking its specific motion, and use the correlation between the system's output with the user's input to determine the selection."

In particular, projects have focused on interactions for public displays. Pursuits by Vidal et al. [41] demonstrated an eye tracking method that, instead of using the direct eye gaze coordinates, is based on the trajectory of raw eye movement compared to the trajectories of objects in the field of view. PathSync by Carter et al. [8] extended the idea of matching trajectories to hand movements, a principle the researchers refer to as rhythmic path mimicry, captured using a depth camera in front of a public display. While previous techniques relied on a single modality using particular body parts (e.g., eyes or hands), TraceMatch [10, 11] used a conventional webcam for movement correlation enabling users to produce matching gestures with any given body parts. WaveTrace [39] extends motion correlation for public displays sensed using the inertial measurement unit available in a smartwatch. The researchers also demonstrated that the same technique could be extended for interactive television and multi-user input on a secondary screen [40].

Overall, motion correlation techniques for large displays offer intuitive, walk-up interfaces, and single- and multi-user input with various body parts. However, they depend on tracking the user while they are stationary and are not specifically designed for everyday wearable computing scenarios. Eye tracking techniques can be extended for use during everyday activities with the availability of mobile eye trackers. Orbits by Esteves et al. [14] enabled hands-free input on smartwatches by matching the smooth pursuit movements of the eyes to the path of a target on the smartwatch screen. Their technique detects whether the user is looking at a target and disambiguates which target. Some of the major disadvantages of eye tracking are that our eyes are used for daily activities and navigation leading to inevitable distractions, sensing is susceptible to ambient interference, and vision-based eye trackers typically require significant power and computation. Dhuliawala et al. [13] further extended the use of smooth pursuit movements, indirectly tracking eye movement captured with a pair of electrooculography (EOG) glasses. EOG glasses account for some of the limitations of vision-based eye tracking devices by eliminating ambient light interference and high computation cost.

In this paper, we present SynchroWatch which is a synchronous gesture system that combines one-handed input with sensing, output stimulus, and computation running live on the smartwatch. Our technique combines ideas behind rhythmic interaction (i.e., periodic input) and synchronous gestures (i.e., output stimuli to inform user input). We adopt many of the same algorithmic approaches from motion correlation work, where the input may or may not be periodic, to analyze how user input matches a given output stimuli. However, our work is based on blinking interface elements and not on movement of interface objects. We contribute a new sinusoidal feature using projection of the magnetometer signal, use cross-correlation between the feature and expected result to adjust for user lag, and finally correlate the two signals to determine object selection. To summarize, SynchroWatch focuses on the periodic motion of the thumb in response to output stimuli from the smartwatch and measures the relative change in the signal over time. In the remainder of the paper, we describe our implementation and suggest avenues for further development in this space.

3 SYNCHROWATCH CONCEPTUAL DESIGN

We describe the sync gesture, interaction design, interface and feedback design, and hardware setup used in our technique in the following subsections.

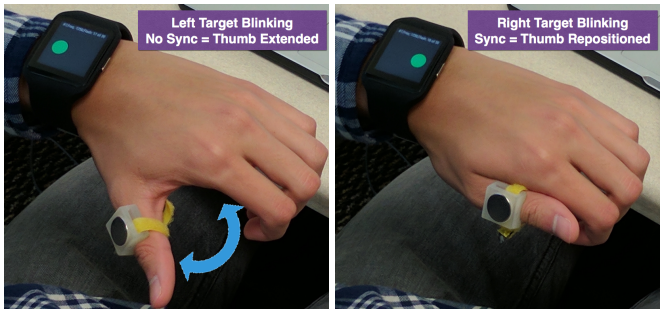

Fig. 2. Visual representation of the SynchroWatch gesture, including a thumb extension (left) and reposition (right). A thumb reposition, closest to the hand, indicates a 'select' or 'sync'. In this figure, there are two blinking targets on the screen - left and right - which are blinking out of phase at the same frequency. The user is 'syncing' to the blinking target on the right side of the screen to select it.

3.1 User Interface and Feedback Design

The SynchroWatch user interface consists of two circular blinking targets of size 96x96 pixels each, a standard pixel resolution for smartwatch application icons. The blinking targets are flashing out of sync as seen in Figure 2. In this example, we focus on the target on the right side of the screen as the sync target. Figure 2 (left) shows only a target on the left side of the screen turned on, while the right blinking target which is the target of interest is turned off. In this figure, the thumb is extended in the 'no sync' position. Figure 2 (right) shows the thumb in a 'sync' position with the target of interest (circle on the right) turned on, indicating the user's intent to 'sync' to the right target.

To select a target, the user will 'sync' to that target by moving their thumb toward the hand, as explained in the following subsection. The blinking circles are colored blue to indicate possible targets. Once the user has synced to a particular target, the blinking circle turns green based on the correlation score to signify a sync gesture has been detected for that target. The size of the blinking circles targets can be smaller (10 x 10 pixels) than shown in the figure and placed alongside interface elements for integration into applications, reducing clutter in the interface. Alternatively, the on-screen interface elements (e.g., application icons, buttons, etc.) themselves could be animated to blink or fade in/out at a known frequency.

3.2 Gesture Design

The synchronous gesture consists of a thumb extension (away from the hand) and reposition (toward the hand). A visual representation is shown in Figure 2. We define a thumb reposition to be a 'sync' action and a thumb extension as a 'no sync' state. The gesture design is inspired by a classic mouse click and release. The reposition of the thumb, when closest to the hand, also provides tactile feedback with the thumb touching the side of the index finger. Intuitively, these two thumb states initially suggest the design of a binary controller with two on-screen targets. We describe additional possibilities on how to extend the interaction in the following section.

3.3 Interaction Design

The overall interaction possibilities for SynchroWatch span a range of possible approaches. Please see the video figure and demonstration applications section for more details.

SynchroWatch allows users to select an on-screen target by matching the motion of their thumb to the flashing of that on-screen target. SynchroWatch can be used with a single target on the screen, representing selection of a

discrete command. This interaction style might be used in situations such as dismissing a text notification, or to return to the watch's home screen.

SynchroWatch can be used with two targets on the screen, allowing the user to select one of two discrete command alternatives. This interaction might be used in scenarios such as choosing whether to accept a phone call or send it to voice mail. The SynchroWatch algorithm could be expanded to allow selection from among more than two on-screen targets but the focus of this work is on binary selection.

The input expressiveness can also be extended through combinations with other gestures. For example, SynchroWatch can be combined with gestures such as wrist flips to switch between modes (e.g., pairs of blinking targets) or alternatively perform other commands. We describe an example of this interaction idea in the video figure and demonstration application section.

Synchronous selection could also be combined with additional gestures to parameterize the action that is selected; this technique could be used to create 'continuous' actions. For example, a 'sync and hold' approach, by keeping the thumb repositioned once a target is synced, allows the user to extend the duration of the selected command without the need to continue syncing. This method could be used to continuously increase volume, or scroll position, based on how long the user's thumb is in the 'hold' position.

3.4 Hardware Setup

We use a Sony Smartwatch 3 SWR50 for prototyping and evaluation. The device is based on a Quad ARM A7 1.2 GHz processor. We capture raw data from the smartwatch motion sensors, sampling the magnetometer, accelerometer, and gyroscope, at the highest sampling rates of 100 Hz, 200 Hz, and 200 Hz, respectively. The system samples the accelerometer and gyroscope to support *post hoc* analysis. Our technique leverages magnetic sensing with a thumb-worn ring. We focus on magnetic sensing as it enables, given the presence of a strong magnet nearby, a robust sensing modality. We designed a 3D-printed mount for a magnet which sits on top of the user's thumb, and fastened the ring using a thin adjustable velcro strap. We use a rare earth Neodymium (N35) magnet disk with dimensions 16mm X 3mm for data collection and evaluation. The ring is designed as a prototype and can ultimately be fashionably designed to fit a user's lifestyle and thumb size.

4 SYNCHROWATCH ALGORITHM AND IMPLEMENTATION

4.1 Detector Pipeline

The SynchroWatch detector based on the algorithm and pipeline described below is designed and implemented as a Java library. The same detector can be used for both offline and online analysis. The detector runs on a laptop for offline analysis of magnetometer data collected during an experiment session (see Section 5). The detector is also incorporated as part of a standalone Android application running only on the Sony SmartWatch 3 and provides live feedback on syncing correlation (i.e., how well is the user syncing?) and direction (i.e., which binary target is the user syncing to?). Live feedback is shown to the user and provided by changing the color of the intended object based on the correlation score.

4.2 Algorithm Design

The pipeline consists of four main phases: user interface stimuli and feedback, feature generation using deltas vector projection, lag adjustment using cross-correlation, and the detection of syncing and direction. Figure 3 shows a high-level diagram flow of the pipeline.

4.2.1 Simple Harmonic Motion Model. We can approximate the synchronous extension and reposition of the thumb at a given frequency as a simple harmonic motion. More specifically, we ask the user to perform a periodic motion with the extension and reposition of the thumb. The resulting minimum-effort motion profile will loosely

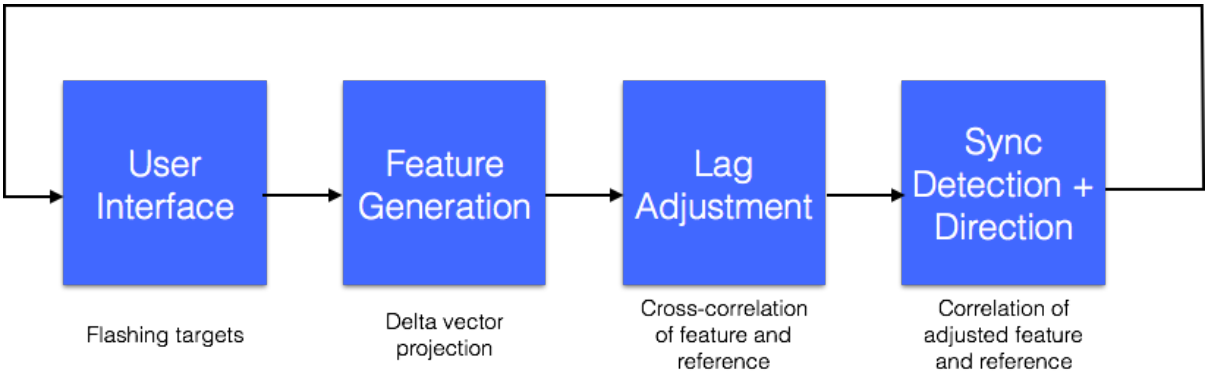

Fig. 3. Visual flow of the SynchroWatch pipeline, including user interface, feature generation, lag adjustment using cross-correlation, and detection of syncing action and direction.

resemble a sinusoidal wave: an acceleration, then constant rate, then rapid deceleration. Given the position of the smartwatch on the wrist and the magnetic ring on the thumb, the magnetometer will return its lowest magnitude reading when the thumb is extended and the magnet is farthest from the magnetometer. When the thumb is repositioned, the magnet is closest to the magnetometer, and the reading will be highest. The magnetometer cannot distinguish between the magnetic field caused by the magnetic ring versus the magnetic field of the earth or stray fields in the environment. It simply returns a 3D vector that represents the sum of these fields. However, as the thumb moves back and forth, the end of the magnetometer vectors describe a line (see Figure 4) upon which the sinusoidal motion occurs (much in the same way a pendulum's motion is modeled by sinusoidal movement upon a line). If the user turns while performing the gesture, the magnetic field of the earth can distort this line. In practice, though, most human body movement is significantly slower than the approximately 1 second per cycle we have defined as the required gesture, and the change in magnetic field strength still roughly defines a line.

4.2.2 Magnetometer Sensor Data Pre-Processing. The magnetometer vectors are sampled at 100 Hz. The detector filters and down samples the vectors down to 10 Hz to remove signal noise and reduce the computation required for live processing on the smartwatch. The magnetometer vectors within every incoming 100 ms (i.e., 10 Hz) window are averaged. The timestamp for the averaged vectors is set to 50 ms (i.e., center of the window) away from the first magnetometer vector timestamp in the window. The timestamp of the left blink and right blink events are also passed to the detector.

4.2.3 Transformation Using Deltas Vector Projection. The motion of the thumb is captured by the magnetometer, each measurement resulting in a 3D sample. Let $H(k) \in \mathbb{R}^3$ be the magnetometer sample at time t_k. The magnitude of the measurement $M(k) \in \mathbb{R}$, given in (1), represents the magnetic field strength (with addition of noise) at each sample time. This magnitude is heavily influenced by the distance of the magnet to the watch, as well as by other surrounding magnetic fields.

$$M(k) \stackrel{\Delta}{=} \left\|H(k)\right\|_2 = \sqrt{H(k)_x^2 + H(k)_y^2 + H(k)_z^2} \tag{1}$$

Intuitively, one might consider using the magnitude of the magnetometer signal to sync with the reference signal (e.g., sine wave) as the natural thumb movements bring the magnet closer to the watch. However, based on pilot studies, this situation is not always the case: some results show a negligible variance in the magnitude,

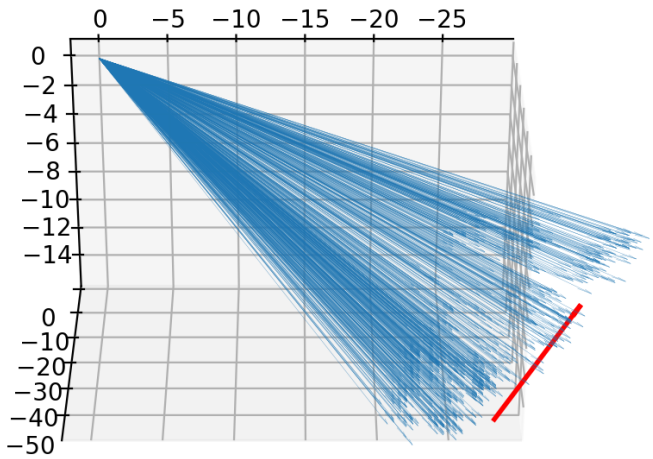

Fig. 4. As the user performs the required sync gesture, the resulting 3D magnetic field vectors (i.e., blue lines) detected by the magnetometer sweep through a vector (i.e., red line) upon which the motion appears sinusoidal.

while the individual measurements in $H(k)$ exhibit a clear sinusoidal-like pattern (see Figure 5). In addition, no particular axis of the measurements (or a predefined linear combination) can be used exclusively, as the plane of the finger motion, and the ring magnet direction, may vary with the user.

We introduce a new feature that accounts for the dynamic and repetitive nature of the synchronous gesture and does not depend on knowing or finding the exact plane of thumb movement. We define a 1D coordinate system based on the full sync reading cycle (i.e., the known timing of a left blink, right blink, left blink). The mean of all 3D magnetometer vectors sampled within the cycle are averaged and result in a magnetometer *mean vector* for the cycle. The two vectors farthest away (calculated using angular distance) from the mean are assigned to be the magnetometer vectors — *left vector* and *right vector* — that occurred at the first left blink and right blink in that cycle. The left and right vectors are subtracted resulting in a *basis vector* that is a vector line across the window. The vector of the current magnetometer reading (during a cycle) is subtracted from the mean vector to find the current *delta vector*. The current delta vector is projected onto the basis vector, resulting in a sinusoidal waveform as the user's thumb oscillates, as shown in Figure 5. Overall, the vector projection using deltas significantly outperforms the magnitude across participants.

Once the system has computed the deltas, the data is detrended using a simple regression of all the points in the window to minimize effects of wrist motion and the earth's magnetic field changing around the user. The relative changes in the magnetic field induced by the thumb movement are strong enough to preserve a measurable signal, even while walking. The detector interpolates the sensor data with a constant time interval to account for uneven sensor arrival times, a commonly known challenge of using a non real-time operating system such as Android.

4.2.4 Determining Input Response Lag Using Cross-Correlation. The goal is to find the maximum correlation between the synthesized deltas feature and a reference waveform. The reference waveform is tuned to match the frequency of the blinking targets on screen, which can be adjusted as necessary.

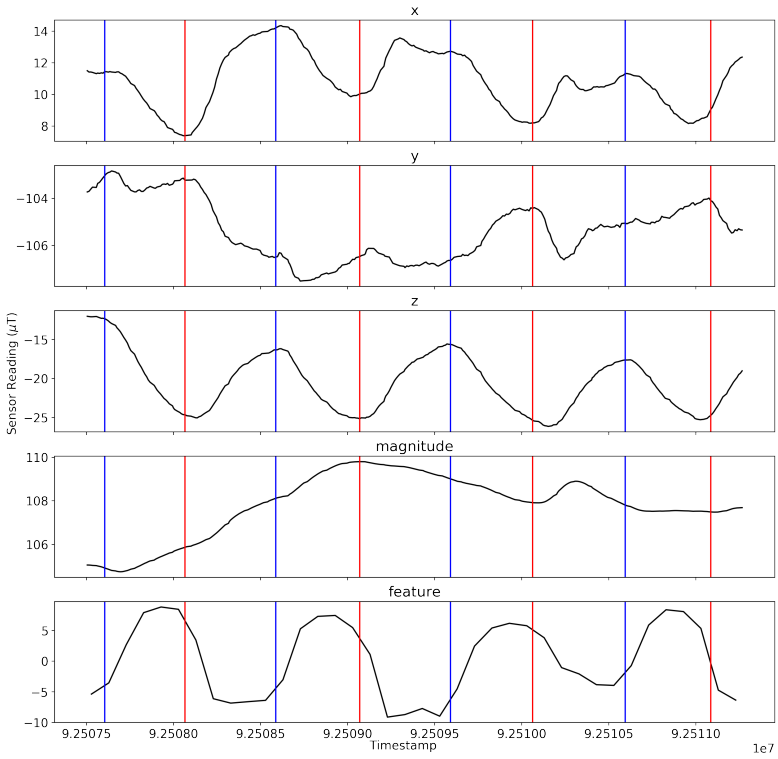

Fig. 5. Visualization of raw magnetometer data for a sync trial during a sitting activity. The vertical lines indicate when the on-screen targets blinked – the red line corresponds to the left target and the blue line to the left target blinks. The plotted lines correspond to the 3-dimensional components of the magnetometer: x, y, z. The magnetometer magnitude and delta vector projection feature are shown. The desired outcome is for the peak of the feature to be closely centered about the right target blink (i.e., red vertical line) when selecting to that target.

When generating the reference waveform, it is necessary to account for input response lag to achieve the highest accuracy. This latency refers to the time delta between the visual stimulus and the thumb in 'sync' state, and is caused by a variety of factors, most notably human reaction time caused by limitations in perception and motor response. Research has reported that human response times of haptic and visual stimuli is age-, gender-, and fatigue-dependent [31]. During pilot studies, we observed an average lag approximately 200 milliseconds between the arrival of the thumb at the sync position and the target blink. Thus, it is crucial to dynamically auto-calibrate and time-shift the reference waveform during usage, depending on the observed input lag, to ensure maximum correlation.

A window of 1.5 seconds, determined empirically based on post hoc analysis from our evaluation, is used to cross-correlate between the feature and reference wave signal. The cross-correlation identifies similarity at different lags relative to one another. We find the optimal time shift to apply to the generated reference wave (see Figure 6).

The maximum or minimum of the cross-correlation output indicates the time where the signals are best aligned. In other words, the time delay between the two signals is determined by the argument of the maximum, or arg max of the cross-correlation, where f is the delta feature and g is the reference waveform:

$$\tau_{\text{delay}} = \arg\max_{t}((f \star g)(t)) \tag{2}$$

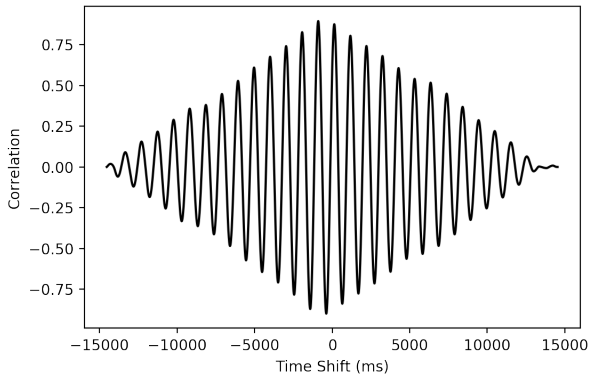

Fig. 6. Output of the cross-correlation between a reference sinusoidal waveform and the magnetometer delta feature signal sampled from the watch. As an example, cross-correlation shown here is calculated over a 15 second window resulting in less than a second lag. The time delta between zero and the location of the argmax determines the suggested time shift for a particular trial.

4.2.5 Time-Shifted Sine Wave Correlation Using Deltas. Once the time-shifted reference wave is aligned with the sensor data, the detector performs the correlation between the two signals to detect the synchronous gesture and target. We use correlation as a statistical measure of dependence between the two sets of data. We leverage the commonly used Pearson product-moment correlation coefficient, r, which measures the linear dependence between two variables X and Y. In this equation, n is the number of samples, x_i and y_i are the single samples indexed with i, and $\bar{x}$ and $\bar{y}$ are the sample means. See Equation 3. The total correlation between the two variables is measured as a value from -1 to 1, where $r = 1$ denotes total positive correlation, $r = 0$ is no correlation, and $r = -1$ is total negative correlation.

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{3}$$

The correlation coefficient is used to gauge the similarity of the sensor readings and the reference wave and to determine whether the user input is synchronized with either one of the on-screen stimuli. Figure 7 shows a small window of the delta vector projection with the overlay of a sinusoidal reference waveform. The system is able to detect (a) whether a 'sync' event is being performed by exceeding a correlation threshold and (b) which blinking element is being targeted based on the directionality of the coefficient (±1). Figure 8 shows an example of increasing correlation as a user is syncing to the right target blinking at 1 Hz.

Our choice of algorithm using correlation is informed by 1) the need to measure the shift in time of our sensor signal and effectively compare to a known expected reference signal, and 2) the desire to build a 'walk-up' interface that does not require extensive user training or supervised machine learning. We considered alternative algorithms, such as DTW or training-based learning algorithms, but they did not satisfy our design requirements for the system architecture and interface/gesture design.

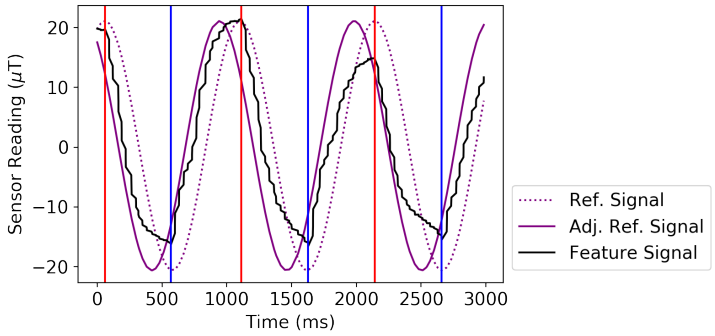

Fig. 7. A short window of data highlighting the delta projection feature overlaid with the reference sinusoidal waveform. Frequency is 0.8 Hz or period of 1250 ms. The dotted purple line indicates the reference sinusoid signal centered around the flashes (i.e., red vertical line is a right target blink, blue vertical line is a left target blink). The black line represents the feature based on the user's thumb movement and appears early. The solid purple line is the reference sinusoid signal after time-shifting using cross-correlation.

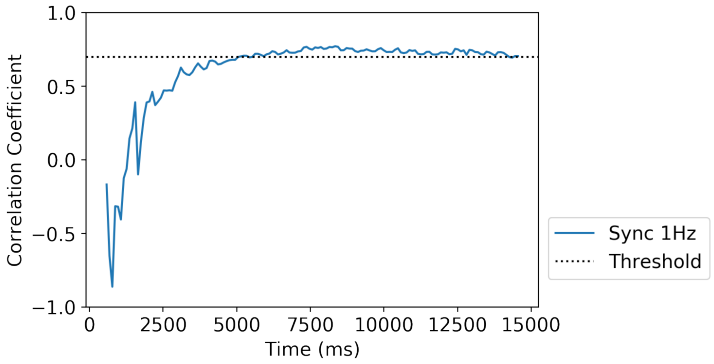

Fig. 8. Example correlation over time graph for a user syncing to the right target at 1 Hz.

5 EVALUATION I - OFFLINE ASSESSMENT OF SYNCHROWATCH SYNCING CORRELATION

The goal of the first study was to evaluate participants' offline performance syncing at three different frequencies (i.e., 0.8 Hz, 1 Hz, 1.25 Hz) during three different activities (i.e., browsing, walking, relaxing). We conducted a technical and user evaluation of the interaction technique with an unmodified Sony Smartwatch S3 in the usability lab of our institution. Upon arrival, participants signed a consent form. After the study, participants completed a questionnaire to assess their experience during the study. The study lasted approximately one hour per participant.

5.1 Participants

We recruited eight participants from our institution (6 male, 2 female, ages 18-30) via word of mouth. All participants are students. Seven participants indicated not wearing a smartwatch regularly. Participants received $10 compensation for their time.

5.2 Design and Experimental Setup

Participants wore the watch and magnetic ring on their left hand. To begin, researchers performed a demonstration of the gesture to be performed. Participants familiarized themselves with the data collection application during a practice round.

The study was designed to capture sensor data in a semi-controlled environment. Participants were asked to perform the following tasks naturally and were not given any special instructions. During the evaluation, they were prompted using haptic feedback on the smartwatch, using a random Poisson delivery time of one minute, to perform a sync event.

When prompted to perform the sync gesture, they raised their arm and watch to line of sight and completed the sync. For the purposes of the evaluation, all participants were asked to sync to the right on-screen target. The targets were flashed at different frequencies (1.33 Hz, 1 Hz, 0.8 Hz) which correspond to different blink times (750 ms, 1000 ms, 1250 ms) for 15 cycles after the prompt. We define a full cycle as the flashing of the left and right targets. The size of the two targets was 96 x 96 pixels, each centered on the left and right half of the screen.

The three distraction tasks (see Figure 9) assigned to participants were:

- Browsing: The participants sat at an office desk in front of a laptop. They were asked to search online and type out the responses to a list of 20 random trivia questions (e.g., What is the capital of Denmark?, What is the unemployment rate in your country?, What is 33% of 23,857?, How many inches are in 5 miles?, etc.). The task was designed to simulate everyday use of a laptop computer including using the trackpad, typing text, and pressing number keys.
- Walking: Participants walked along an indoor walking path. The path included walking in straight lines and random turns. At least one researcher walked alongside participants and engaged them in conversation while the task was taking place.
- Relaxing: The participants sat in front of a laptop and were given the option to choose their favorite sitcom or movie to watch. Participants were asked to sit naturally and rested their hands on their lap for the duration of the exercise while the video played.

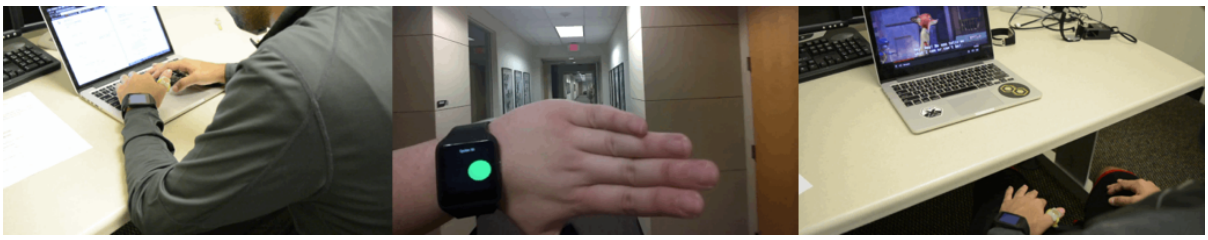

Fig. 9. User study distraction tasks: browsing on the computer, walking, and relaxing watching videos. The hand was raised to chest level during the evaluation. For demonstration only, the hand is shown raised to eye level in the walking condition.

Data was recorded for the entire session from beginning to end of each distraction task, which included approximately 4-5 minutes of sync data and 15 minutes of noise data (in which no intentional sync events were performed) for each of three activities. In total, we collected 1 hour of sensor data for each of the 8 participants. During each activity, the system prompted participants 5 times for each frequency for a total of 15 rounds. The order of the tasks and the blink frequencies were randomized for each participant. In summary, the user study includes 8 participants x 3 distraction tasks x 3 frequencies x 5 repetitions per frequency x 15 sync cycles per round for a total of 5400 sync cycles.

5.3 Results

The steps of the detection algorithm include the calculation of the deltas vector projection feature, determining input lag using cross-correlation with a sinusoidal waveform, and correlation between the two signals to determine a similarity coefficient. We present results for each of these phases and some additional analyses.

5.3.1 Determining Input Response Lag. We visually demonstrate that the delta vector projection can outperform magnitude (see Figure 5) in the first stage of the algorithm. Here, we present results for the second stage determining the input lag between the user's response and visual stimuli using cross-correlation. On average across participants and all 360 sync trials (8 participants x 3 activities x 3 frequencies x 5 rounds), human reaction time was approximately 429 milliseconds. The average input response lag per frequency is as follows: 226ms/1.33Hz, 422ms/1Hz, 641ms/0.8Hz. Participants reported the 1 Hz frequency as comfortable and our correlation over time results indicated the highest correlation at this frequency.

5.3.2 Correlation Over Time with a Growing Window. We use the input response lag time computed per trial to auto-calibrate and adjust the reference waveform. We then calculate the correlation over time to the deltas vector projection (as explained in the algorithm section). The goal of the correlation over time is to demonstrate separability between intended sync and null class data.

The correlation over time is calculated by running the detection technique on growing windows of data from 0.1 sec in 0.1 sec intervals until the full 15 cycles of data are included. We achieve an average correlation coefficient between 0.5 and 0.6 between the participant's gesture and the sync signal (see Figure 10).

We also compute the correlation over time using the delta vector projection against the noise or null dataset. Specifically, we chose 100 random samples from the null class of the appropriate window size, again growing the window size from 0.1 sec to full 15 cycles of data in 0.1 sec intervals. We achieve an average correlation coefficient between 0.1 and 0.2 for the null class data, which is significantly lower than the correlation when the user is synchronizing.

As seen in Figure 10, the correlation coefficients between synchronized gestures and normal movement become well separated within three seconds, suggesting that a detector can be made that detects a synchronized gesture relatively quickly.

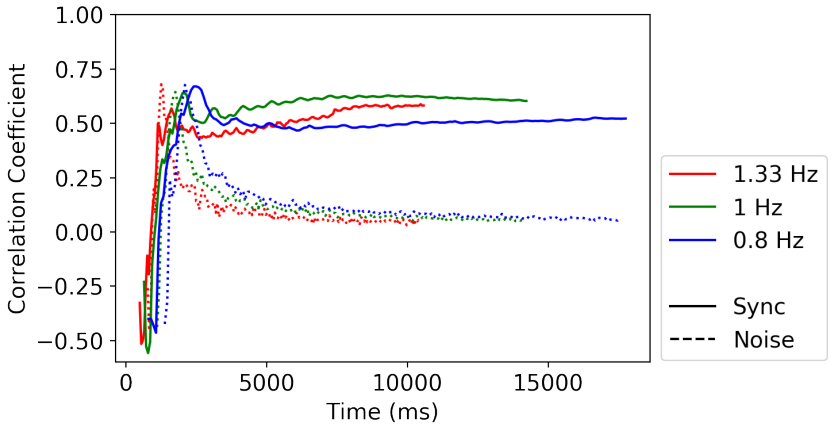

Fig. 10. Correlation over time using delta vector projection for signal (solid lines) and noise data (dotted lines). The colors indicate the blinking frequency (red = 1.33 Hz, green = 1 Hz, blue = 0.8 Hz). The correlation is calculated over 15 cycles for different frequencies, explaining the different time lengths for each line.

5.3.3 Correlation with a Sliding Window. Based on the offline analysis of our correlation over time with a growing window, we wanted to determine which window size would be ideal for correlation to inform our real-time implementation. We empirically determined a sliding window of 1.5 seconds for cross-correlation to be suitable for achieving high correlation between the feature and known signal. We use the 1.5 second window for cross-correlation in a real-time implementation of the algorithm, evaluated in the following section. The result of introducing a sliding window also meant that the correlation threshold could be set higher, as correlation values upwards of 0.9 were more likely to occur in smaller window sizes. We determined empirically that a correlation threshold of 0.9 was a conservative threshold to use for our real-time implementation.

5.4 Discussion

We generate the correlation over time graphs between the sensor readings and a reference sine wave to determine the length of time needed to adequately detect a user syncing. In general, the goal is to find the least amount of time needed to converge past a detection boundary. Based on analysis, we observed that an increased window length leads to higher correlation with an asymptotically increasing pattern. Initially, the correlation calculation is seemingly unstable with the presence of a peak prior to stabilizing. We hypothesize the peak is likely due to participants moving their wrist in synchrony with the stimulus when randomly prompted during data collection. This initial wrist movement of raising the watch to the field of view is visible in the raw magnetometer data (see Figure 5) and causes unpredictable results during the initial seconds of detection. A way to mitigate this behavior is to set appropriate correlation thresholds and empirically set a waiting time before triggering.

QuickDraw by Ashbrook et al. [4] investigated the effect of placement and user mobility on the time required to access an on-body interface. The researchers found that a wrist-mounted system was significantly faster to access than a device stored in the pocket or mounted on the hip and reported access time of 2.787 sec. Access time includes reaching for the watch and responding to an alert. While the repetitive nature of the SynchroWatch gesture may appear slow, the technique is only slightly slower than accessing the touchscreen with the non-watch hand. It takes approximately 3 seconds to achieve a stable correlation coefficient above 0.5 for the three frequencies, after raising the arm and beginning to sync.

One of the advantages of the SynchroWatch technique is that it automatically compensates for the user input response lag and does not require calibration. In addition to the evaluation using a Sony Smartwatch 3 device, we also successfully deployed and ran the algorithm on an LG G Watch device. The technique is agnostic to the orientation of the sensor and the sensor topology inside the smartwatch, as it leverages relative changes in the magnetic field across axes with the delta vector projection feature.

6 EVALUATION II - LIVE COMPARISON OF SYNCHROWATCH VS. TOUCH

The goal of the second study is to evaluate the online performance of participants using two interactions — syncing vs. touch swiping. The systems in this study use sensor data from the device for online analysis and live feedback is provided for syncing and swiping. Sensor data is recorded for both the swiping application (e.g., touch location, gyroscope, accelerometer, magnetometer) and the syncing application (e.g., accelerometer, magnetometer).

6.1 Syncing vs. Touch Swiping

During the evaluation, participants were interrupted with a visual/haptic notification on the smartwatch during the evaluation activity (i.e., sitting or walking). Their task was to dismiss the notification as quickly as possible using two distinct interactions — syncing and touch swiping.

The first interaction to dismiss a notification in this evaluation used SynchroWatch by 'syncing right', which only requires the use of the watch hand and is the focus of this work. A correlation threshold trigger of 0.9 is

used to trigger syncing. The color of the sync target changes color (blue to green) based on the correlation score and sync direction. A higher correlation results in the syncing target turning green. Once a trigger above the threshold was detected, a text display indicating a sync detected was shown.

The second interaction was touch 'swiping right' using the non-watch hand to swipe a target on the left side of the screen to the right side of the screen (Figure 11A). Once the icon being swiped had reached the other side of the screen and was within the touch buffer zone (Figure 11B), the target changed colors to green as feedback of task completion to the user (Figure 11C). A text display indicated a swipe detected was shown once the participant released the target inside the target zone. This action follows the current mode of interaction on many commercial watches and phones today to slide to unlock the device or answer a phone call.

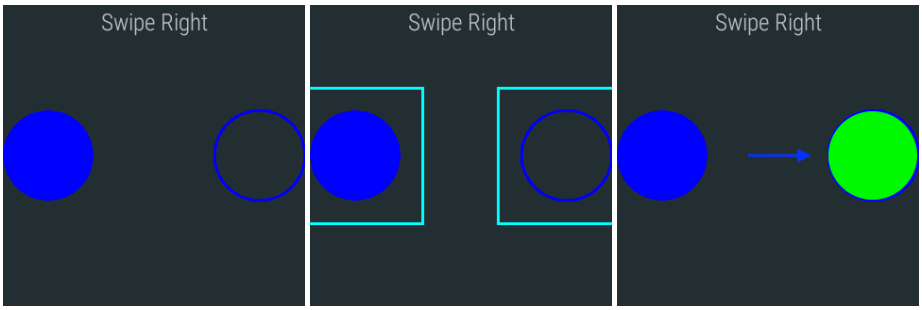

Fig. 11. A) Screenshot of the swipe application interface presented to the user with two selection targets/icons. B) Swipe interface showing relaxed touch target areas for selecting the right or left targets to swipe. C) Swipe right completed by the user turns the right side circle green.

6.2 Participants

We recruited eight participants from our institution (5 male, 3 female, ages 18-28). All participants were students. The study lasted approximately two hours. Participants received $10 compensation for their time. The majority of participants reported not regularly wearing a smartwatch or other wearable device. Three participants did indicate multiple years of musical experience (e.g., playing an instrument or reading music), which may facilitate 'syncing' to a particular beat or stimulus.

6.3 Design and Experimental Setup

We conducted a technical evaluation of the interaction techniques (sync vs. swipe) with an unmodified Sony SmartWatch 3 in the usability lab of our institution. Upon arrival, participants signed a consent form. After the study, participants completed a questionnaire to assess their experience during the study.

The within-subjects study was designed for two conditions (sitting and walking) and two interactions (syncing and touch swiping). A 4x4 Latin square design was used to counterbalance the sitting/walking and syncing/swiping. Two practice sessions were completed, followed by two evaluation sessions. Participants were randomly assigned to a row of the Latin square for the first practice session. The same order of the Latin square was used for the second session. A new randomly assigned row of the Latin square was used for the first evaluation. The same order of the square was used for the second evaluation.

Participants wore the smartwatch and magnetic ring on the left hand. To begin, researchers played an introduction and training video to participants, demonstrating each of the interactions and the visual/haptic notification they would receive prompting them to complete the task. Participants were asked to dismiss the notification

using syncing or swiping. Participants familiarized themselves with the data collection applications for syncing and swiping by performing (while sitting) at least 5 sync examples and 5 swipe examples.

The study was designed to capture sensor data in a semi-controlled environment. During the study and for both interactions, the participant was asked to keep their arm and hands on their lap (while sitting) and down by the side of their hip (while walking). Participants were asked to perform the following tasks naturally and were not given any special instructions. For both applications, a haptic vibration was delivered to the user randomly using a Poisson delivery time of 10 seconds prompting them to raise their arm to chest level and line of sight. The interface controls for syncing and swiping appeared on-screen as soon as the device vibrated.

For the purposes of the evaluation, all participants were asked to sync and swipe to the right on-screen targets. For syncing, the targets were flashed at 1 Hz corresponding to a blink period of 1000 ms. We define a full cycle as the flashing of the left and right targets. A correlation threshold of 0.9 was used to trigger a sync event. The application prompted users to sync for 10 cycles after the prompt, as well as swipe right 10 times. The size of the two targets was 96 x 96 pixels, each centered vertically on the far left and right half of the screen.

The two distraction tasks assigned to participants were:

- Walking: Participants walked along a random indoor walking path. The path included walking in straight lines and random turns. At least one researcher walked alongside participants and engaged them in conversation, while the task was taking place.
- Sitting: The participants sat in front of a laptop and were given the option to choose their favorite sitcom or movie to watch. Participants were asked to sit naturally and rested their hands on their lap for the duration of the exercise while the video played.

Data was recorded in individual files during each of the sync and swipe tasks in each of the distraction tasks. The length of a syncing session with 10 syncs was approximately 3.5 minutes. The length of a swiping session with 10 swipes was approximately 2.5 minutes. Noise data was collected for the interim period when the participants are waiting to be prompted to perform the task.

In total, we collected sensor data for each of the 8 participants across two practice sessions and two evaluation sessions for sitting and walking. The entire study with each participant lasted approximately two hours.

For each sync session, the system prompted participants 10 times to sync. In summary, the user study included 8 participants x 2 distraction tasks x 1 frequency x 10 sync repetitions per frequency x 10 sync cycles per round for a subtotal of 1600 sync cycles. Across two practice and two evaluation sessions, we collected a total of 640 sync events.

For each swipe session, the system prompted participants 10 times to swipe. In summary, the user study included 8 participants x 2 distraction tasks x 10 swipe repetitions for a subtotal of 160 swipe events. Across two practice and two evaluation sessions, we collected a total of 640 sync events.

6.4 Results

6.4.1 Syncing - Correlation with a Sliding Window. The correlation over time is calculated live by running through the detection pipeline described above. A correlation coefficient is output every 100 ms for a 1.5 second running window of sensor data, after the cross-correlation is calculated on that same window of data. The correlation coefficient increases as a function of time reaching between 0.8 and 0.9 for the syncing event (see Figure 12). Also shown are normalized values for the acceleration vector including gravity. The acceleration data shows the movement of the arm raise from a rest state to syncing state. The time between intention and action can be described as the time it takes the user to raise the arm to provide input, which can be inferred from this accelerometer data.

6.4.2 Syncing - Accuracy vs. Time vs. Thresholds. To further analyze whether the synchronized gesture might be detected accurately versus normal movement, we plot a 3D graph of correlation coefficient vs. time vs.

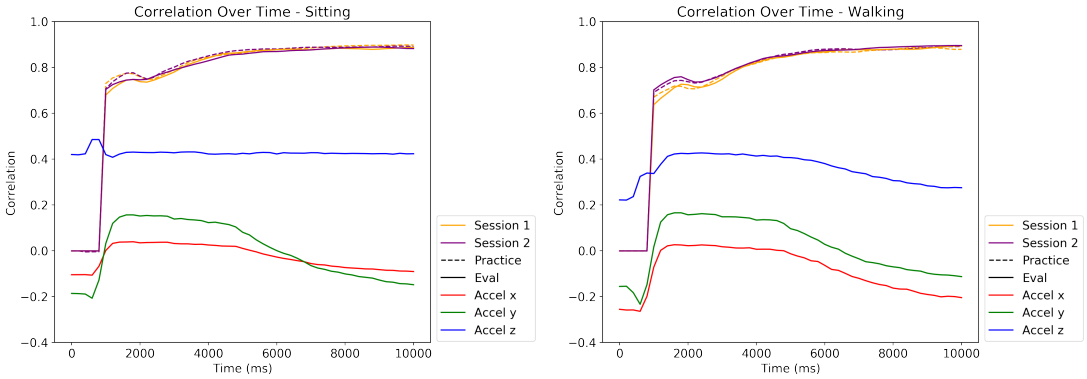

Fig. 12. Correlation over time using delta vector projection for sitting (left) and walking (right). Results for practice and evaluation sessions are averaged across all users. Figure also includes the normalized acceleration vector of the wrist including gravity showing when the arm is raised for syncing (within the first 1.5 to 2 seconds).

accuracy. Using different thresholds, we run the detector post hoc on the data where the participants are trying to synchronize their thumb movements to the graphics and the interim period where they are just sitting or walking. The results are shown in Figure 13. Accuracy is defined in this graph as the average between the percent accuracies for detecting true positives and true negatives. True negatives are defined as 1 - false positive rate. False positives are unintentional triggers (i.e., exceeding the correlation threshold) using the null set when no intentional input is being performed. The optimal results for sitting and walking indicates syncing is possible within 4 seconds at a correlation threshold of approximately 0.8. In post hoc analysis, we perform a maximum likelihood analysis of the accelerometer gravity vector during syncing and non-syncing to capture the orientation of the hand. We use this information as a filter to minimize false positives. In other words, when the user is syncing the hand is placed at eye level in front of the user resulting in a certain orientation of the hand. While walking, the user rests their hand at the side of their body which results in another hand orientation which should not trigger syncing.

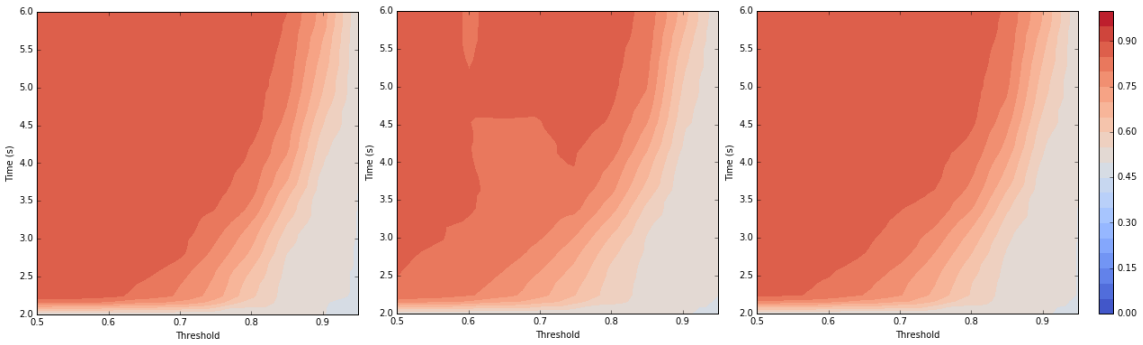

Fig. 13. 3D plot of threshold vs. time vs. accuracy over time for syncing during various activities: sitting (left), walking (middle), and sitting and walking combined (right).

6.4.3 Syncing and Swiping - Time to Trigger. The time between intention and action and the interaction time have been previously suggested as useful metrics to assess the value of an interaction [37]. Microinteractions are quick actions that typically last only a few seconds [3].

The time to completion is one metric of interest we evaluate for both interactions (i.e., syncing and swiping). For syncing, the time to sync depends on the correlation coefficient threshold used to trigger. During the live evaluation, a correlation coefficient of 0.9 is used as the trigger threshold. A 0.9 coefficient was observed on the upper end of the syncing scores during pilot testing. Additionally, a conservative threshold results in higher times to trigger. However, it also enables further analysis post hoc by reducing the threshold to be less conservative and reassessing the time to trigger.

The time to trigger includes the time required by the user to raise the arm from rest to chest level and complete the swipe. Thus, the total time to trigger includes the time between intention and action (i.e., setup time) and the interaction time.

As observed in the correlation over time graphs (see Figure 12), no correlation coefficient is reported for the first window of data. The earliest time the system could potentially trigger is set to 1.5 seconds to avoid triggering while the person is raising the arm.

Overall, it took participants approximately 5.5 seconds to complete the sync event on average with a correlation coefficient of 0.9. By reducing the coefficient to 0.8, we observe that sync times can be reduced to under 4 seconds. For swiping, the time to trigger was on average 2.1 seconds across both sitting and walking. The time to trigger includes the time required by the user to raise the arm from rest to chest level and complete the swipe. See Figures 14 and 15.

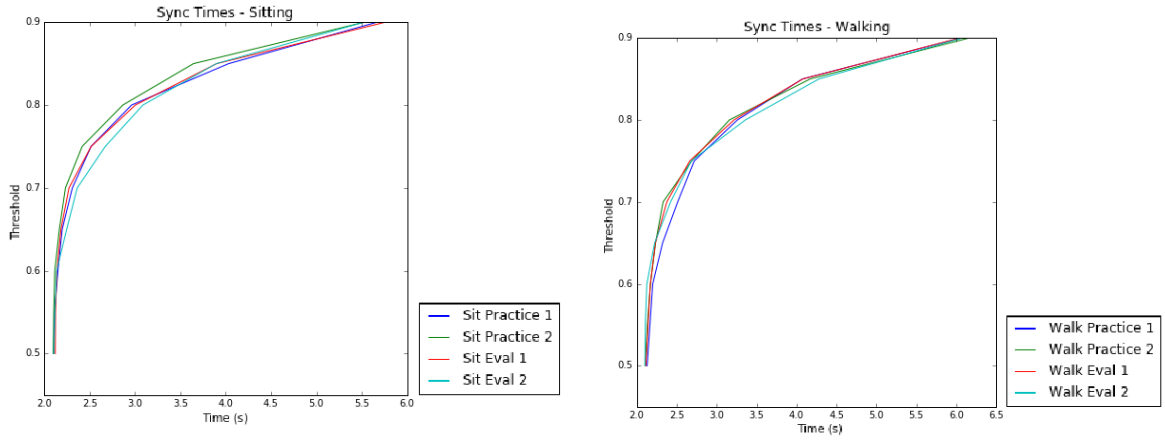

Fig. 14. Figure showing time to sync for sitting (left) and walking (right) at various correlation thresholds. Results for practice and evaluation sessions are averaged across users.

6.5 Discussion

We generate the correlation over time graphs between the sensor readings and a reference sine wave to determine the length of time needed to adequately detect a user syncing. In general, the goal is to find the least amount of time needed to converge past a detection boundary. During the first evaluation, we demonstrated based on offline data that participants achieved separable correlation over time results. The goal of the second evaluation was to empirically demonstrate syncing accuracy with a live system and compare the time to trigger between touch swiping using the non-watch hand.

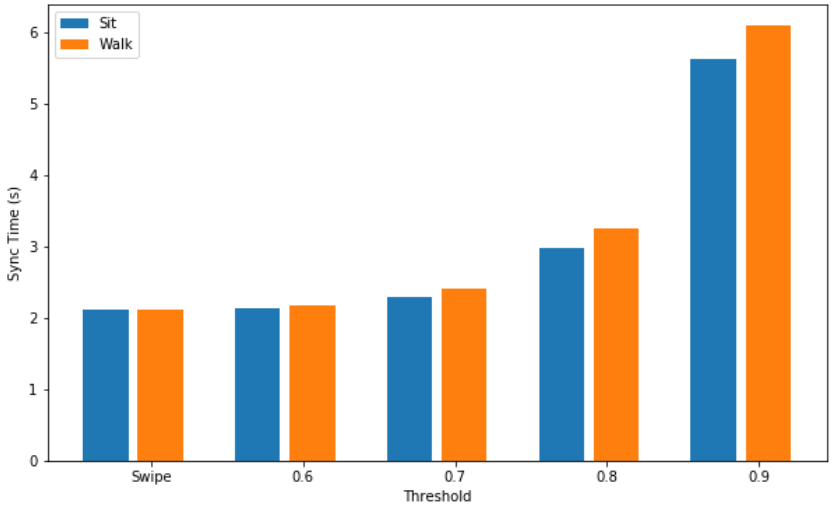

Fig. 15. Summary of the times to trigger for swipe/sync and sitting/walking. Sync times are reported for various correlation thresholds (0.6 to 0.9).

Overall, the SynchroWatch technique is slightly slower than accessing the touchscreen with the non-watch hand. During the second study, it took approximately 2.1 seconds for participants to complete the swipe action. Using a conservative correlation threshold of 0.9, it took participants approximately 5.5 seconds to sync while sitting and 6 seconds to sync while walking. However, by setting the coefficient between 0.7 and 0.8, offline analysis post hoc indicates that a sync time of approximately 3 seconds is possible to achieve while sitting and walking. We also tested workload as measured by the NASA TLX and syncing required slightly more workload than swipe for both sitting and walking, but the key advantage is that our technique is one-handed.

In the notification-response scenario, the user would receive a haptic notification at the wrist. The user would raise their arm and bring the screen within their field of view. By motivating the interaction through an ecologically valid scenario, we would not begin detection algorithm until the user is looking at the watch (which can be indirectly inferred by them raising the arm). The synchronization blinking targets would appear at that point for the user to begin 'syncing'.

We introduced a simple pose detection approach in our post hoc analysis to determine if the user's hand is in position to 'sync' before enabling the system, and then detecting whether they are 'syncing'. We used pose detection to also eliminate false positives and calculate the accuracy vs. threshold vs. time graphs. The threshold-based pose detector using maximum likelihood could be developed to be more robust with further tuning or training data, resulting in higher accuracies. Another approach to determining if the user is ready to 'sync' is to use the ON_WAKE gesture callback provided by the smartwatch once the screen turns on and the user has performed the arm lift gesture.

Two additional aspects that require further exploration are the impact of hand size of the wearer and how they perform syncing. We observed a few cases of magnetometer readings with low amplitude (i.e., weak signals) from users that impacted the calculation of the deltas projection feature and system accuracy. We suspect it's influenced by two factors: 1) the hand size which may influence the distance between ring and the watch, and 2) the degree of extension vs. reposition of the thumb. Subtle movements are more challenging to capture than gross motions (in particular while walking) and may result in a detrimental user experience.

7 DEMONSTRATION APPLICATIONS

We have developed several example applications that help illustrate the usefulness of SynchroWatch. The applications highlight the use of SynchroWatch for response and command gestures with corresponding blinking interface elements. See Figure 16. Our applications are partly inspired by prior motion correlation demonstrations [14], but our interfaces are adapted for rhythmic correlation.

Dismiss Phone Call: Quick access to view and respond to notifications is one of the most compelling features of a smartwatch. After a notification prompts the user, a *response* gesture can be used to select between two actions. In the phone call application, two flashing icons to answer the call or send to voicemail are blinking out of phase on the screen. The user declines the call discretely by syncing with their thumb to the flashing decline icon, without even needing to lift the non-watch hand. This demonstrates the use of SynchroWatch for binary selection.

Text Viewer: The second application uses SynchroWatch as a *command* gesture to scroll through an article. We have two flashing targets at the top and bottom of the screen. Syncing to the bottom target scrolls the page down, while syncing to the top target scrolls up. In the current implementation, the user must continuously 'sync' to continue scrolling. In a future implementation, it may be possible to modify the detection approach to enable continuous scrolling after syncing. In other words, the user could extend and reposition their thumb until the application begins to scroll, then either maintain the thumb extended to stop or repositioned to continue scrolling. This facilitates a 'sync and hold' interaction and does not require continuous syncing that may lead to fatigue.

Music Player: We increase the expressivity of SynchroWatch beyond binary target selection by introducing a wrist flick gesture. The wrist flick flips between two (or more) pairs of blinking controls. In the music player application, we present two pairs of blinking icons toggling between a vertical and horizontal layout, aligned similar to a D-Pad. The top and bottom targets are used to increase and decrease the volume, respectively. The left target is used to play/pause and the right target is used to transition to the next song. Using a separate gesture as a mode switch allows quickly toggling through a series of targets, which may be useful in other multi-target applications such as a messaging app (e.g., star, delete, etc.) or an application launcher.

Fig. 16. Figures highlighting three demonstration applications: answering/dismissing a phone call, scrolling through a text view, and controlling a music player.

8 LIMITATIONS AND FUTURE WORK

There are a number of avenues we could explore to extend the capabilities of SynchroWatch. Allowing more than two targets is an obvious way to increase the expressivity of the technique. However, there is a balance between the number of moving targets visible on the small screen and the usability with the increasing distraction of additional targets. One approach is to have multiple targets blinking in parallel at unique frequencies. Multiple reference waveforms would have to be generated, complicating the detection algorithm and potentially increasing false positives. Another approach is to use a round-robin blink sequence (i.e., same frequencies presented out of phase). A round-robin approach, used in the current implementation, may prove less distracting to the user with only a single target blinking at a time. However, increasing the number of targets will also decrease their observable frequency potentially slowing down the interaction. An advantage of the technique is that the reference sinusoidal waveform could be generated using any known frequency.

SynchroWatch currently supports syncing between two blinking targets that are out of phase at three frequencies (1.33 Hz, 1 Hz, 0.8 Hz). Some participants expressed difficulty syncing to a target blinking at 1.33 Hz, especially during the walking task. One idea is to adjust the frequency of the blinking controls depending on the activity context. For example, typical walking frequencies range from 1-2 Hz [7]. Based on motion sensor data from the smartwatch, the system could determine the ideal blinking frequency to increase the separation from the current activity (e.g., walking). Alternatively, a simpler approach is to allow the user to specify the desired blinking frequency. In this case, muscle memory may enable eyes-free interaction when prompted.

Another way to enable eyes-free interaction is to leverage other non-visual feedback channels for the given stimuli. For example, haptic feedback on the smartwatch or using a bone conduction actuator on a pair of smart glasses. Acoustic feedback can also be provided using a pair of earbuds. Visual feedback may also be extended by disaggregating the input and output feedback location (i.e., watch screen). Visual blinking controls could be displayed using the screen or a pair of LEDs in a head-up display, or on a nearby wall display in a smart environment. In this case, the smartwatch device is used only for sensing, and timing between devices should be managed appropriately.

The recognition results reflect the system's performance in a semi-controlled environment with syncing during three tasks: relaxing while watching videos, actively browsing on the computer, and walking. Given the goal of achieving an interface that operates while the person is moving, the technique leverages magnetic sensing with a thumb ring. However, a practical disadvantage is that some users may be disinclined to wear a passive magnetic ring. One advantage of the rhythmic correlation technique is that it can be extended to operate with the accelerometer and gyroscope. We observed that the thumb extension and reposition gesture induced slight movement in the accelerometer and gyroscope but not sufficient for detection, without using gross movements. However, an alternate gesture design (e.g., raising/lowering or rotating the wrist) matching a given blink frequency may be used without the need of a passive ring. This exploration remains future work. Past work has explored a similar idea for mobile devices [5]. Another caveat is that we intuitively suspect that using the accelerometer and gyroscope will require appropriate signal separation to be robust to global motion (e.g., walking).

9 CONCLUSION

SynchroWatch is an interaction technique for one-handed gestures on smartwatches that relies on *synchrony* between blinking on-screen controls and motions performed by the user. In particular, the system focuses on capturing continuous thumb extension and reposition movements matching on-screen flashes at a given frequency. The technique uses magnetic sensing with a passive magnetic ring on the thumb. The relative motion of the thumb ring induces a measurable vector change in the magnetic field around the smartwatch, which is correlated to the known stimuli frequency. We evaluate the performance of the technique using time-shifted

correlation with a reference signal based on a harmonic motion model. The technique is calibration-less, resulting in user- and device-independence. The technique is robust to noise and global motion enabling one-handed input while walking. We evaluate the technique initially offline during three tasks with varying degrees of hand and finger movement: walking, browsing on the computer, and sitting while watching a movie. We then evaluate the technique during a second study with an online version of the system, providing users feedback on their performance. We also compare the selection of binary targets with SynchroWatch versus the current mode of interaction using touch swiping. Results demonstrate that SynchroWatch is a viable input modality requiring only one hand, while touch swiping is faster to complete a selection but requires the use of the non-watch hand. Finally, we conclude and demonstrate opportunities enabled by SynchroWatch with various potential use cases.

10 ACKNOWLEDGEMENTS

This work is generously supported by a Google PhD Fellowship and a Google Faculty Research Award. We thank our reviewers and user study participants for their contributions.

REFERENCES

[1] Daniel Ashbrook, Patrick Baudisch, and Sean White. 2011. Nenya: Subtle and Eyes-free Mobile Input with a Magnetically-tracked Finger Ring. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2043–2046. https://doi.org/10.1145/1978942.1979238

[2] Daniel Ashbrook and Thad Starner. 2010. MAGIC: A Motion Gesture Design Tool. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 2159–2168. https://doi.org/10.1145/1753326.1753653

[3] Daniel L. Ashbrook. 2010. *Enabling Mobile Microinteractions*. Ph.D. Dissertation. Georgia Institute of Technology, Atlanta, GA, USA. AAI3414437.

[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. https://doi.org/10.1145/1357054.1357092

[5] Peter Bennett, Stuart Nolan, Ved Uttamchandani, Michael Pages, Kirsten Cater, and Mike Fraser. 2015. Resonant Bits: Harmonic Interaction with Virtual Pendulums. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '15)*. ACM, New York, NY, USA, 49–52. https://doi.org/10.1145/2677199.2680569

[6] Daniel Boland and Roderick Murray-Smith. 2013. Finding My Beat: Personalised Rhythmic Filtering for Mobile Music Interaction. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '13)*. ACM, New York, NY, USA, 21–30. https://doi.org/10.1145/2493190.2493220

[7] Agata Brajdic and Robert Harle. 2013. Walk Detection and Step Counting on Unconstrained Smartphones. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*. ACM, New York, NY, USA, 225–234. https://doi.org/10.1145/2493432.2493449

[8] Marcus Carter, Eduardo Velloso, John Downs, Abigail Sellen, Kenton O'Hara, and Frank Vetere. 2016. PathSync: Multi-User Gestural Interaction with Touchless Rhythmic Path Mimicry. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3415–3427. https://doi.org/10.1145/2858036.2858284

[9] Ke-Yu Chen, Kent Lyons, Sean White, and Shwetak Patel. 2013. uTrack: 3D Input Using Two Magnetic Sensors. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 237–244. https://doi.org/10.1145/2501988.2502035

[10] Christopher Clarke, Alessio Bellino, Augusto Esteves, and Hans Gellersen. 2017. Remote Control by Body Movement in Synchrony with Orbiting Widgets: An Evaluation of TraceMatch. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3 (Sept. 2017), 45:1–45:22. https://doi.org/10.1145/3130910

[11] Christopher Clarke, Alessio Bellino, Augusto Esteves, Eduardo Velloso, and Hans Gellersen. 2016. TraceMatch: A Computer Vision Technique for User Input by Tracing of Animated Controls. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*. ACM, New York, NY, USA, 298–303. https://doi.org/10.1145/2971648.2971714

[12] Andrew Crossan and Roderick Murray-Smith. 2006. Rhythmic Interaction for Song Filtering on a Mobile Device. In *Haptic and Audio Interaction Design (Lecture Notes in Computer Science)*. Springer, Berlin, Heidelberg, 45–55. https://doi.org/10.1007/11821731_5

[13] Murtaza Dhuliawala, Juyoung Lee, Junichi Shimizu, Andreas Bulling, Kai Kunze, Thad Starner, and Woontack Woo. 2016. Smooth Eye Movement Interaction Using EOG Glasses. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction (ICMI 2016)*. ACM, New York, NY, USA, 307–311. https://doi.org/10.1145/2993148.2993181

[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. https://doi.org/10.1145/2807442.2807499

[15] Emilien Ghomi, Guillaume Faure, StÃĪphane 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. https://doi.org/10.1145/2207676.2208579

[16] Jun Gong, Xing-Dong Yang, and Pourang Irani. 2016. WristWhirl: One-handed Continuous Smartwatch Input Using Wrist Gestures. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 861–872. https://doi.org/10.1145/2984511.2984563

[17] Timo Gotzelmann and Pere-Pau Vazquez. 2015. InclineType: An Accelerometer-based Typing Approach for Smartwatches. In *Proceedings of the XVI International Conference on Human Computer Interaction (Interaccion '15)*. ACM, New York, NY, USA, 59:1–59:4. https://doi.org/10.1145/2829875.2829929

[18] Anhong Guo and Tim Paek. 2016. Exploring Tilt for No-touch, Wrist-only Interactions on Smartwatches. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. ACM, New York, NY, USA, 17–28. https://doi.org/10.1145/2935334.2935345

[19] 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. https://doi.org/10.1145/1622176.1622199

[20] 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)*. ACM, New York, NY, USA, 149–158. https://doi.org/10.1145/964696.964713

[21] Scott E. Hudson, Chris Harrison, Beverly L. Harrison, and Anthony LaMarca. 2010. Whack gestures: inexact and inattentive interaction with mobile devices. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction (TEI '10)*. ACM, New York, NY, USA, 109–112. https://doi.org/10.1145/1709886.1709906

[22] Sungjae Hwang, Myungwook Ahn, and Kwang-yun Wohn. 2013. MagGetz: Customizable Passive Tangible Controllers on and Around Conventional Mobile Devices. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 411–416. https://doi.org/10.1145/2501988.2501991

[23] Hamed Ketabdar, Mehran Roshandel, and Kamer Ali YÃijksel. 2010. Towards Using Embedded Magnetic Field Sensor for Around Mobile Device 3D Interaction. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '10)*. ACM, New York, NY, USA, 153–156. https://doi.org/10.1145/1851600.1851626

[24] Ju-Whan Kim, Han-Jong Kim, and Tek-Jin Nam. 2016. M.Gesture: An Acceleration-Based Gesture Authoring System on Multiple Handheld and Wearable Devices. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 2307–2318. https://doi.org/10.1145/2858036.2858358

[25] 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. https://doi.org/10.1145/2642918.2647356

[26] Gierad Laput, Robert Xiao, and Chris Harrison. 2016. ViBand: High-Fidelity Bio-Acoustic Sensing Using Commodity Smartwatch Accelerometers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 321–333. https://doi.org/10.1145/2984511.2984582

[27] Rong-Hao Liang, Kai-Yin Cheng, Chao-Huai Su, Chien-Ting Weng, Bing-Yu Chen, and De-Nian Yang. 2012. GaussSense: Attachable Stylus Sensing Using Magnetic Sensor Grid. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 319–326. https://doi.org/10.1145/2380116.2380157

[28] Kent Lyons. 2016. 2D Input for Virtual Reality Enclosures with Magnetic Field Sensing. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers (ISWC '16)*. ACM, New York, NY, USA, 176–183. https://doi.org/10.1145/2971763.2971787

[29] 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. https://doi.org/10.1145/2380116.2380134

[30] Sylvain Malacria, Eric Lecolinet, and Yves Guiard. 2010. Clutch-free Panning and Integrated Pan-zoom Control on Touch-sensitive Surfaces: The Cyclostar Approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 2615–2624. https://doi.org/10.1145/1753326.1753724

[31] Annie W. Y. Ng and Alan H. S. Chan. 2012. Finger Response Times to Visual, Auditory and Tactile Modality Stimuli. *Lecture Notes in Engineering and Computer Science* 2196, 1 (March 2012), 1449–1454. https://doaj.org

[32] 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. https://doi.org/10.1145/2556288.2557138

[33] Aman Parnami, Apurva Gupta, Gabriel Reyes, Ramik Sadana, Yang Li, and Gregory D. Abowd. 2016. Mogeste: A Mobile Tool for In-Situ Motion Gesture Design. In *Proceedings of the 8th Indian Conference on Human Computer Interaction (IHCI '16)*. ACM, New York, NY, USA,

35–43. https://doi.org/10.1145/3014362.3014365 bibtex: parnami_mogeste:_2016.

[34] 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. https://doi.org/10.1145/2470654.2466192

[35] Gabriel Reyes, Dingtian Zhang, Sarthak Ghosh, Pratik Shah, Jason Wu, Aman Parnami, Bailey Bercik, Thad Starner, Gregory D. Abowd, and W. Keith Edwards. 2016. Whoosh: Non-voice Acoustics for Low-cost, Hands-free, and Rapid Input on Smartwatches. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers (ISWC 2016)*. ACM, New York, NY, USA, 120–127. https://doi.org/10.1145/2971763.2971765

[36] Jaime Ruiz and Yang Li. 2011. DoubleFlip: A Motion Gesture Delimiter for Mobile Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2717–2720. https://doi.org/10.1145/1978942.1979341

[37] T. Starner. 2013. Project Glass: An Extension of the Self. *IEEE Pervasive Computing* 12, 2 (April 2013), 14–16. https://doi.org/10.1109/MPRV.2013.35

[38] 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 (April 2017), 22:1–22:35. https://doi.org/10.1145/3064937

[39] David Verweij, Augusto Esteves, Vassilis-Javed Khan, and Saskia Bakker. 2017. WaveTrace: Motion Matching Input Using Wrist-Worn Motion Sensors. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, New York, NY, USA, 2180–2186. https://doi.org/10.1145/3027063.3053161

[40] David Verweij, Vassilis-Javed Khan, Augusto Esteves, and Saskia Bakker. 2017. Multi-User Motion Matching Interaction for Interactive Television Using Smartwatches. In *Adjunct Publication of the 2017 ACM International Conference on Interactive Experiences for TV and Online Video (TVX '17 Adjunct)*. ACM, New York, NY, USA, 67–68. https://doi.org/10.1145/3084289.3089906

[41] Melodie 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. https://doi.org/10.1145/2493432.2493477

[42] Hongyi Wen, Julian Ramos Rojas, and Anind K. Dey. 2016. Serendipity: Finger Gesture Recognition Using an Off-the-Shelf Smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3847–3851. https://doi.org/10.1145/2858036.2858466

[43] 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. https://doi.org/10.1145/2807442.2807500

[44] 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. https://doi.org/10.1145/2556288.2557017

[45] Cheng Zhang, Abdelkareem Bedri, Gabriel Reyes, Bailey Bercik, Omer T. Inan, Thad E. Starner, and Gregory D. Abowd. 2016. TapSkin: Recognizing On-Skin Input for Smartwatches. In *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces (ISS '16)*. ACM, New York, NY, USA, 13–22. https://doi.org/10.1145/2992154.2992187

Received November 2016; revised August 2017; accepted October 2017