Sound of Motion: Real-time Wrist Tracking with A Smart Watch-Phone Pair

Submission #1570667010 to IEEE INFOCOM 2021

***Abstract*—Proliferation of smart environments entails the need for real-time and ubiquitous human-machine interactions through, mostly likely, hand/arm motions. Though a few recent efforts attempt to track hand/arm motions in real-time with COTS devices, they either obtain a rather low accuracy or have to rely on a carefully designed infrastructure and some heavy signal processing. To this end, we propose SoM (*Sound of Motion*) as a lightweight system for wrist tracking. Requiring only a smart watch-phone pair, SoM entails very light computations that can operate in resource constrained smartwatches. SoM uses embedded IMU sensors to perform basic motion tracking in the smartwatch, and it depends on the fixed smartphone to act as an "acoustic anchor": regular beacons sent by the phone are received in an irregular manner due to the watch motion, and such variances provide useful hints to adjust the drifting of IMU tracking. Using extensive experiments on our SoM prototype, we demonstrate that the delicately engineered system achieves a satisfactory wrist tracking accuracy and strikes a good balance between complexity and performance.**

***Index Terms*—Tracking, acoustic signal, smartwatch, smartphone, human-machine interface**

I. Introduction

Though *untethered Human-Machine Interface* (uHMI) has a long development history under the video game industry, it is mostly dominated by technologies that require an infrastructure support, e.g., computer vision and voice recognition [1]–[4]. Recently, we start to witness an increasing demand on infrastructure-free uHMIs due to the proliferation of smart environments. Powered by successes in both *Internet of Things* (IoT) and *Virtual/Augmented Reality* (VR/AR) (e.g., Oculus Rift [5] and HoloLens [6]), we are immersed into environments full of smartness anytime anywhere. To better interact with these new environments, we urgently demand ubiquitously deployable uHMIs free of infrastructure constraints.

There are indeed a few attempts in designing uHMIs based only on *Commercial Off-The-Shelf* (COTS) devices, such as smartphones and smartwatches [7]–[9]. These proposals pioneered in using COTS devices for tracking the hand/arm motions, so as to drive a uHMI through the digitized motion traces (e.g., playing virtual golf). However, both AAMouse [7] and CAT [9] require an acoustic transmitter array (an infrastructure not always available) to achieve cm-level tracking accuracy, which may confine their applicability to smart environments. Also, the heavy computation load incurred by intensive signal processing tasks makes them infeasible to existing smartwatches. ArmTrak [8], on the contrary, relies solely on a smartwatch and certain domain-knowledge driven priors to track arm motions. Its online version is computationally light by relying on prior point cloud and orientation information, but may not guarantee the tracking accuracy and smoothness even under strong priors (e.g., fixed shoulder position).

Fig. 1. Sound-driven wrist tracking with a smart watch-phone pair.

In this paper, we intend to develop a system striking a good balance between CAT and ArmTrak: it relies on COTS devices available to most human users (namely a pair of smartwatch and smartphone) to enable ubiquitous deployability, and it achieves a satisfactory real-time wrist tracking accuracy by using the fixed smartphone as an "acoustic anchor" to correct the IMU-driven motion tracking in the smartwatch (Fig. 1). This novel yet plausible watch-phone pair concept faces two major challenges posed by the resource limited smartwatch: firstly, even the most up-to-date smartwatch cannot handle the heavy signal processing algorithms (e.g., the 44k-point FFT for AAMouse/CAT [7], [9] and the synchronization procedure for CAT) while catching up with the IMU sensing running at 100Hz level. Secondly, the information offered by a single anchor is too little to assist the motion tracking in the conventional manner, such as Kalman or particle filtering [10], [11] commonly used for robotics.

To tackle these challenges, our SoM (or Sound of Motion) system fixes the smartphone (e.g., on a table) and lets it emit inaudible sound tones at regular intervals. Observed by the smartwatch in motion, these intervals may vary due to the changing distance between the pair. Such a *Motion-induced Time Difference of Arrival* (MTDoA) naturally avoids synchronization between the pair, while generating ranging information for correcting the IMU-based motion tracking. In order to precisely detect the arrival time of a tone, SoM adopts the very lightweight Sliding Goertzel DFT algorithm [12], whose constant time complexity allows it to be fit into the limited time frames and the resource constrained smartwatch. Moreover, SoM innovatively utilizes the ranging information provided by MTDoA to opportunistically rectify the IMU-based tracking, instead of performing constant corrections in a filtering manner. In summary, we make the following major contributions in designing SoM:

- The novel concept of real-time wrist tracking with a smart watch-phone pair; it has the potential to enable ubiquitously deployable uHMIs.
- A sound-driven MTDoA scheme that delivers useful error-correcting observations on wrist tracking, while avoiding complicated synchronization between the pair.
- A suit of lightweight signal processing procedures to synthesize the IMU/sound sensing results, delicately tailored to stringent time frames and computational resources.
- A prototype that demonstrates a reasonable wrist tracking accuracy with only a pair of smartwatch and phone.

SoM is not aiming to compete with AAMouse [7] and CAT [9] in tracking accuracy. Instead, it showcases that uHMIs can be conveniently deployed with smartwears commonly available to human users. In the following, we shall first survey the related literature to motivate our design in Sec. II. Then we present the core technologies of SoM on both acoustic ranging and motion tracking in Sec. III and IV. We summarize the system architecture and execution pipeline is Sec. V. The system evaluation is reported in Sec. VI, before concluding the paper in Sec. VII.

II. Related Work

Literature related to motion tracking is vast. Computer vision has been widely adopted in both academia [13]–[15] and industry [1]–[3], but it is an intrinsically different methodology that is computationally intensive and requires line-of-sight. Another emerging trend is RF-enabled tracking [16]–[18], yet it entails a heavy signal processing pipeline and a purposely arranged infrastructure. Device-free acoustic tracking [19]–[21] is possible but offers a limited detection range and is susceptible to multipath interference. Exisiting proposals on gesture recognition [8], [21]–[27] can only identify a small number of patterns and thus fails to track free-form wrist/arm motions. Therefore, we omit these marginally related literature but rather focus on device-based acoustic sensing and IMU technologies for motion tracking in the following.

A. Device-based Acoustic Tracking

Acoustic or ultrasonic signals are popular media for ranging, since their *Time of Flight* (ToF) can be readily measured by normal clocks with MHz precision. Nevertheless, the real challenge lies in the precise synchronization between a pair of sender and receiver. Earlier proposals such as Cricket [28] use a concurrent RF signal to perform synchronization, but COTS devices lack of specially designed hardware to ensure instantaneous timestamping for the arrivals of both radio and ultrasonic signals. Guoguo [29] deploys multiple acoustic beacons and sophisticated signal processing techniques to get around the synchronization issue, yet the system architecture is too heavy to serve continuous tracking.

To avoid instantaneous timestamping, BeepBeep [30] requires a pair of devices to mutually transmit acoustic signals; the timestamps of receptions are obtained through cross-correlations and then exchanged using a side channel (e.g., WiFi). Unfortunately, the design may not work in high-mobility systems due to the large computation delay [31]. With a more complicated software structure and Doppler-shift compensation, SwordFight [31] is able to support ranging under 2m/s moving speed, but its sophisticated design may never fit into a low-end mobile device such as a smartwatch. CAT [9] eliminates the need for mutual transmission by *Frequency Modulated Continuous Waveform* (FMCW), at the cost of a further increased overall computational load in terms of signal processing. As follow-ups of CAT, Millisonic [32] supports concurrent tracking of multiple smartphones with a sub-mm tracking accuracy. Moreover, UPS [33] enables regular smartphones to perform ultrasonic tracking by leveraging the nonlinearity effect of microphone.

B. Robotic and Human Tracking

Motion tracking is a central issue of robotics, where Kalman or particle filtering is commonly used with state transitions driven by IMU sensing and observations derived from laser scanner or visual odometery [34]–[37]. Similar mechanisms have been extended and reused for indoor localization and tracking with COTS devices, where innovations are mostly made for deriving observations from (signal) fingerprint matching [38]–[41]. Apparently, none of the aforementioned approaches are applicable to the design of SoM, as mobile devices lack of sophistication in deriving high-accuracy observations (e.g., laser ranging) while the fingerprinting methods work only in a much larger scale.

To overcome the difficulty in acquiring useful observations with a smartwatch, ArmTrak [8] focuses on tracking the elbow whose freedom is rather constrained given the strong assumption that the shoulder is fixed. As a result, given priors on the mapping between watch orientations and potential elbow positions (in the form of point cloud), a motion trace can be treated as a *Hidden Markov Model* (HMM) and thus be "decoded" by Viterbi's algorithm in an offline manner. However, the real-time performance of ArmTrak could not be fully guaranteed as it performs only a weighted average over the point cloud. Therefore, ArmTrak may not satisfy all requirements imposed by a ubiquitous uHMI.

III. Ranging with Acoustic Signal

Ranging is the most commonly used observation for assisting state estimation in motion tracking, so it is also a core to our SoM design. In this section, we first give more justifications on the need for a new design, then we explain the design principles in SoM ranging. We also present certain implementation details in optimizing our design.

A. Motivations

As we have discussed in Sec. II, acoustic ranging appears to be the most suitable choice for a COTS-based design, yet synchronization between the sender-receiver pair should be avoided. There are two typical methods towards a synchronization-free acoustic ranging. One method relies

on the cross-correlations between mutually transmitted signal [30], [31], but it may not work for a watch-phone pair because it is demanding in both hardware and software: i) both sides need to perform transmissions and receptions, ii) a side channel (e.g., WiFi) is required to coordinate between the pair, and iii) correlation computations need to be performed constantly. While many smartwatches do not have a speaker, all these demands cause excessive computation and energy consumptions. Another method requires an acoustic beacon infrastructure (i.e., multiple speakers) [7], [9], confining its ubiquitous deployability. More importantly, as computationally intensive algorithms (e.g., FMCW and FFT) are necessary to process multiple acoustic receptions in real-time, there is virtually no hope for a smartwatch-based implementation. Consequently, we need a novel acoustic ranging method that is lightweight in both hardware and software.

B. SoM Ranging: Differential Perspective

Our idea of *ranging in a differential manner* is rather intuitive (Fig. 2): if a fixed sender transmits a sequence of acoustic tones at a constant interval, the sequence observed by a moving receiver will have dilated (resp. compressed) intervals if the receiver moves away from (resp. towards) the sender. These variances in the intervals are caused by the displacements of the receiver relative to the sender, so that an acoustic tone takes longer (resp. shorter) time to propagate from the sender to the receiver. Denoting an interval variance by $\Delta\tau$, the receiver's displacement can be computed as $c_s \times \Delta\tau$, where c_s is the sound speed. We term such a displacement derived from a variance in interval *radial displacement* hereafter. SoM takes a smartphone fixed on a table as the sender and a smartwatch worn on a user's wrist as the receiver. Therefore, setting the phone to the origin of a global coordinate system, the tone sequence observed by the watch can enable a partial tracking of its position in this coordinate system: integrating over the radial displacement yields the *radial distance*. It is worth noting that no synchronization between the pair is needed, as measuring the radial displacement only requires the knowledge of the tone waveform and the interval.

A commonly adopted method of detecting a tone is to use the known tone waveform to perform cross-correlation with the received signal [30]; this, however, has two drawbacks. On

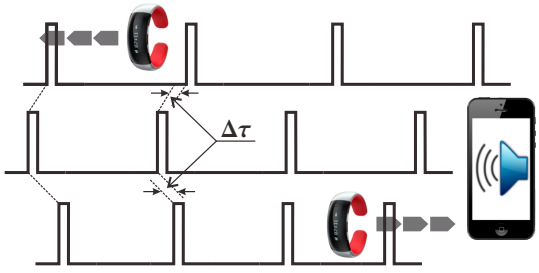


Fig. 2. Differential ranging through Motion-induced Time Difference of Arrival (MTDoA). A fixed smartphone transmits the tone sequence (middle), so a wrist-worn smartwatch moving away from (resp. towards) the phone will observe dilated (resp. compressed) intervals, as shown by the top (resp. bottom) tone sequences.

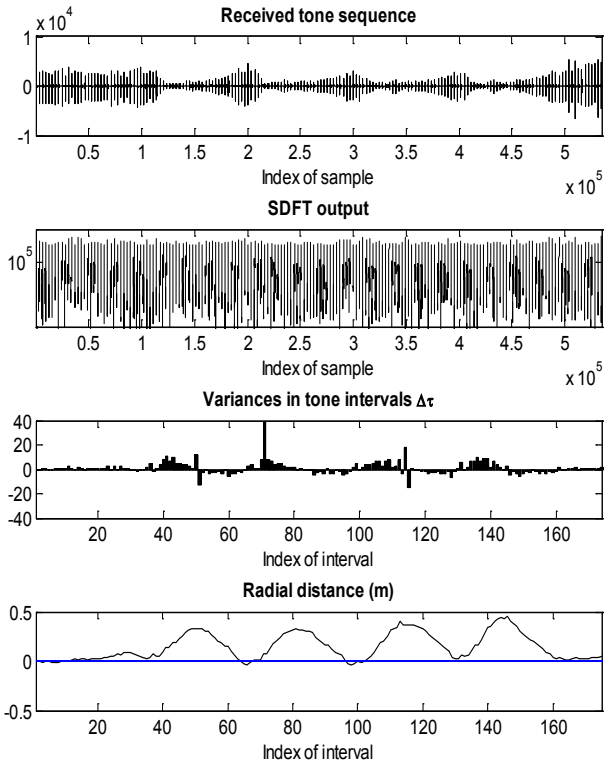


Fig. 3. Tracking radial distance with a tone sequence. We circle the smartwatch 4 times beside the phone, so the tracked radial distance exhibits an expected sinusoid shape.

one hand, the waveform can be so distorted during propagation that the peak of the cross-correlation cannot precisely pinpoint the time of (tone) arrival. On the other hand, the complexity is $\mathcal{O}(N)$ for an N point waveform, which can be a rather heavy burden for a smartwatch. Our observation is that, although the waveform can be heavily distorted in time domain, its energy is still concentrated around the carrier frequency of the tone (we later show that even the Doppler effect can be implicitly handled). Therefore, we perform detection in frequency domain. As the carrier frequency is known, we adopt the Sliding Goertzel DFT algorithm [12] (SDFT hereafter) to only extract the energy concentrated around the carrier frequency. As shown in Fig. 3, the SDFT filter sharply indicates the arrival of tones, resulting in very accurate tracking. Moreover, SDFT has a constant time complexity (actually 3 multiplications and 4 additions [12]) except for the first round.

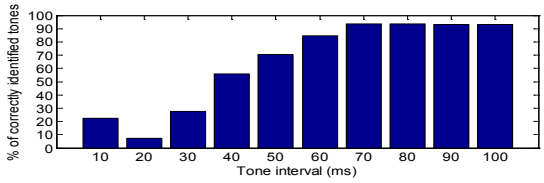
Unlike other sensors, embedded operating systems (such as Android) do not provide timestamps for sound recording, so another challenge is how to determine the (clock) time of tone arrival upon detection. A nice solution to this problem has been proposed by [30] and is adopted by several subsequent proposals (e.g., [31], [42]): as the A/D converter is driven by a 44.1kHz clock and thus the sample interval is a constant 22.7μs, counting the number of samples would serve as a relative timestamp. Nevertheless, as our SoM requires other sensor readings to fully serve the wrist tracking purpose, we need to further align the sound recording with other readings, as will be discussed in Sec. IV-C and V.

Our method is fundamentally different from the Doppler effect based tracking [7], [9], which has to perform a full spectrum FFT (with $N = 44100$ given the 22kHz sound spectrum) and induces an $\mathcal{O}(N \log N)$ time complexity (as opposed to the $\mathcal{O}(1)$ complexity of SDFT). Such a heavy computation load cannot be borne by a smartwatch for real-time tracking. Moreover, estimating frequency shift can hardly be made robust in practice; AAMouse [7] averages the estimations over tones with 5 different carrier frequencies and hence causes an even heavier computation load.

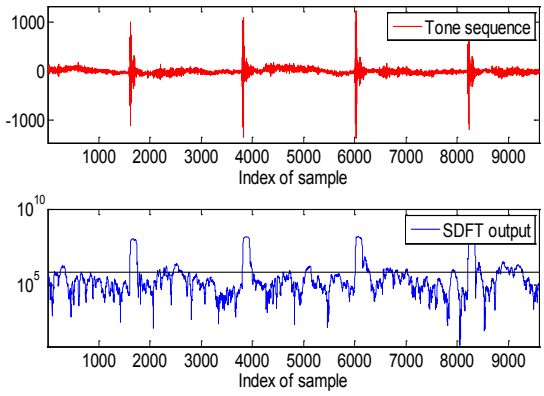
C. System Optimization

The tone interval τ_s in the original transmission is a key parameter to our differential ranging. On one hand, we would like it to be short so that the system can quickly respond to the radial displacement. On the other hand, longer period is more robust against possible noises in the acoustic signal and errors in detecting the arrival time. We test all possible intervals from 10ms to 100ms, with a stepsize of 10ms, and we report the statistics on correctly detected tones in Fig. 4(a). The robustness starts to reduce after τ_s becomes shorter than 60ms due to the reason explained in Fig. 4(b), so we set $\tau_s = 70$ms as a balance between robustness and responsiveness. However, this interval is longer than that (about 5ms) of IMU sensors, so we will handle this mismatch in Sec. IV-C.

Doppler effect may shift the carrier frequency, potentially affecting the detection accuracy of the tone arrival time. Rather than computationally compensating this shift [31], we implicitly handle it by widening the filter window: we perform



(a) Percentage of correctly detected tones vs. τ_s.



(b) $\tau_s = 50$ms: rx signal (red) and SDFT output (blue).

Fig. 4. Selecting a proper tone interval: a balance between robustness and responsiveness. (a) The robustness improves with an increasing τ_s but saturates after $\tau_s = 70$ms. (b) For $\tau_s \leq 50$ms, the interval is so short that a lot of ambient interferences get squeezed inside, making it hard to set a threshold that avoids noise on one hand while detecting the rising edge (not peak) of the tone signal on the other hand.

the 128-point SDFT so that the outcome indicates the total energy within a $\frac{44100}{128} = 344.5$Hz window centered around the carrier frequency. According to [7], a 1Hz shift is equivalent to a speed of 2cm/s at 17kHz, hence our filter window can tolerate a radial speed up to 3.445m/s. Since the arm motion speed for a normal person can hardly exceed 4m/s [43], its projection onto the radial direction (the radial speed) would rarely go beyond 3.445m/s.

One critical observation is that, once a tone is detected, the next arrival time cannot be too soon given the bounded speed of wrist motion; this can further save computational resources for performing IMU-based tracking. Based again on [43], a reasonable displacement during a 100ms interval should be less than 0.5m. Assuming a sound speed of 346m/s at 26°C, 0.5m displacement translates to less than 70 samples. As a result, for the 3087 samples recorded within the 70ms interval, it is impossible that the first 2800 of them contain the next tone, so there is no need to run SDFT over them. Consequently, we can skip 90% of the SDFT computations without compromising the performance.

It is rather unlikely that the 17kHz inaudible frequency exists in the ambient sound, yet the tone sequence sent by our smartphone may be reflected by surrounding objects, causing superfluous receptions during an interval. Whereas the aforementioned "SDFT skipping" helps to partially filter out such receptions, a proper threshold is still necessary for tone detection. Normally, the threshold is set to be sufficiently high to avoid false positive detection, as sporadic false negatives can be handled by skipping a couple of tracking correction opportunities. However, as sound energy attenuates during propagation, the false negative rate will increase with the radial distance d^r, while a fixed threshold often detects a peak rather than a rising edge and hence loses interval detection accuracy, as illustrated by Fig. 5(b). Therefore, we adaptively adjust the threshold according to the power law of d^r:

$$\text{thrd} = \begin{cases} c & d^r \leq 1 \\ c(\beta-1)^{\alpha}(\beta d^r - 1)^{-\alpha} & d^r > 1 \end{cases}, \qquad (1)$$

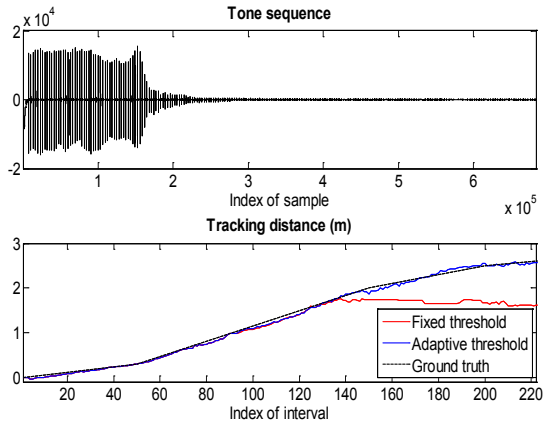
where c is the constant threshold used when $d^r \leq 1$m (adopted in Fig. 4), and $\alpha > 0, \beta \geq 1$ control the attenuation rate. The benefit of an adaptive threshold is shown in Fig. 5: whereas a fixed threshold fails at around 1.8m, an adaptive threshold always achieves a very accurate track of the radial distance.

IV. IMU-driven Motion Tracking

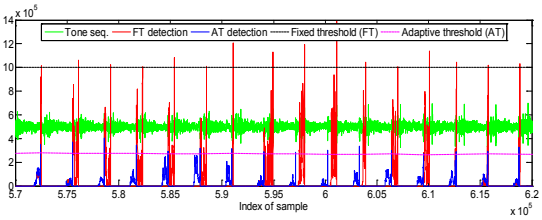
Motion tracking driven by IMU sensing is well understood in robotics, but SoM requires some substantial fine-tuning to meet the needs of wrist tracking with a smartwatch. Our presentation in this section focuses on these innovative aspects.

A. Basics of Kalman Filtering

Kalman filter has been widely used in tracking and navigation, so it might also serve as the basis of SoM. Here we briefly explain the basic procedure of Kalman filtering, so as to establish terminology for further discussions. For a tracking system, the Kalman filter model assumes that the *true state* $\mathbf{s}_k$



(a) Contrasting the tracking effects of two thresholds.



(b) Zooming in the interval detection details.

Fig. 5. Higher robustness and accuracy with an adaptive detection threshold. (a) Adaptive threshold is necessary to maintain the tracking accuracy. (b) When the radial distance goes beyond 1m, a fixed threshold tends to detect the peak of the tone, which causes an unpredictable delay from the rising edge detected by an adaptive threshold.

at time k can be derived from the previous state $\mathbf{s}_{k-1}$ at time $k-1$ by transition $\mathbf{s}_k = A_k\mathbf{s}_{k-1} + \mathbf{w}_k$, where A_k is the state transition matrix and $\mathbf{w}_k \sim \mathcal{N}(0, W_k)$ is the noise assumed to follow a zero mean multivariate normal distribution with covariance W_k. Moreover, an observation $\mathbf{o}_k$ is made on the state at time k: $\mathbf{o}_k = H_k\mathbf{s}_k + \mathbf{v}_k$, where H_k is the state observation matrix and $\mathbf{v}_k \sim \mathcal{N}(0, V_k)$ is the observation noise. If the system is nonlinear, the matrices A_k and H_k are replaced by the Jacobian of the corresponding nonlinear functions. A Kalman filter estimates a state $\mathbf{s}_k$ by two steps: an *a priori* prediction based on state transition and an *a posteriori* correction based on the observation $\mathbf{o}_k$. Basically, the following computations are executed one by one so as to derive an estimation $\hat{\mathbf{s}}_k$ on $\mathbf{s}_k$ from $\hat{\mathbf{s}}_{k-1}$:

$$\hat{\mathbf{s}}_{k|k-1} = A_k\hat{\mathbf{s}}_{k-1}, \quad \mathbf{r}_k = \mathbf{o}_k - H_k\hat{\mathbf{s}}_{k|k-1} \tag{2}$$

$$P_{k|k-1} = A_kP_{k-1}A_k^T + W_k, \quad R_k = H_kP_{k|k-1}H_k^T + V_k,$$
$$K_k = P_{k|k-1}H_k^TR_k^{-1}, \tag{3}$$
$$\hat{\mathbf{s}}_k = \hat{\mathbf{s}}_{k|k-1} + K_k\mathbf{r}_k, \quad P_k = (I - K_kH_k)P_{k|k-1}$$

where the *Kalman gain* K_k in (3) is the key, derived to minimize the mean-square error of the estimation $\hat{\mathbf{s}}_k$.

For a normal IMU-driven tracking system, we have $\mathbf{s}_k = [\mathbf{q}_k^T, \mathbf{v}_k^T, \mathbf{p}_k^T]^T$, where $\mathbf{q}_k$ is the 4D quaternion representing the orientation, $\mathbf{v}_k$ and $\mathbf{p}_k$ are the 3D vectors indicating the velocity and position, respectively. Unfortunately, operating on a 10D vector (thus many 10-by-10 matrices) is infeasible for a resource scarce smartwatch, so we have to substantially reduce the state space. According to Sec. III, the observation derived from the differential ranging is related to $\mathbf{s}_k$ by $o_k = \|\mathbf{p}_k\|_2 - \|\mathbf{p}_{k-1}\|_2$, which is not concerning $\mathbf{q}_k$. This allows us to separate $\mathbf{q}_k$ from the state and estimate it independently. However, the most fatal issue is that we would have to use a scalar observation to correct a high-dimensional state (6D even after removing $\mathbf{q}_k$); this is fundamentally impossible under the filtering framework because the lack of observability [44] makes it hard to identify which component(s) of a state contain(s) noises to be corrected. Therefore, we need a novel solution to apply the rectifications.

B. Orientation Tracking through Opportunistic Calibrations

There are two approaches towards tracking orientation with IMU sensing, namely gyroscope-enabled and accelerometer+magnetometer-enabled. As the former monitors the changes while the latter observes the states, it is possible to construct a specific Kalman filter for orientation tracking [45], but the resulting algorithms are very computationally intensive. An opportunistic calibration approach has been proposed in [46], yet it was designed for tracking slow motions in a large scale (e.g., for human localization and/or navigation), which is very different from wrist tracking for ubiquitous uHMI where the motion can be intense but the scale is rather confined (e.g., around a table). Therefore, our calibration is specifically tailored to wrist tracking.

It is well known that gyroscope-enabled tracking can cause serious drift due to the error accumulations in time. As we cannot hope to get constant observations to correct the tracking in a Kalman filtering manner, the only choice is to "drag" the orientation back to the right track whenever possible, which is often termed *opportunistic calibration*. Based on the physical understanding of the IMU sensors, we know that, when a smartwatch is i) in a relatively steady state and ii) away for a strong magnetic field, accelerometer and magnetometer readings can be used to rather accurately obtain the gravity vector $\mathbf{g}$ and magnetic north $\mathbf{n}$, respectively. Combining these two vectors allows us to estimate the orientation within a global coordinate system whose x-z plane is defined by $\mathbf{n}$ and $\mathbf{g}$. Although $\mathbf{n}$ may be biased indoor, what matters to wrist tracking (given its confined scale) is the stability of $\mathbf{n}$ rather than its correctness. Therefore, we use the following two criteria to declare a *calibration opportunity*: i)

1) $|\|\mathbf{a}\|_2 - g| \leq 0.2\text{m/s}^2$, for an accelerometer reading $\mathbf{a}$ and $g = 9.81\text{m/s}^2$,
2) $\|\mathbf{m}\|_2 \leq 50\mu\text{T}$, for a magnetometer reading $\mathbf{m}$.

We shall not demonstrate the performance of this module here, as the correctness of orientation tracking will be implied by the final performance of wrist tracking.

C. Position Tracking through Opportunistic Calibrations

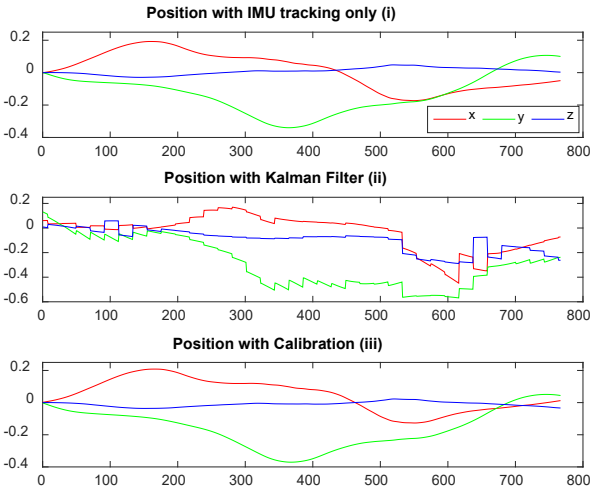
As we explained earlier, Kalman filter would not work because our low dimension observation fails to provide a full observability and hence correction can be wrongly applied to signal (motion) rather than noise (drift). As shown in Fig. 6, using Kalman filter can lead to a tracking outcome worse than

solely relying on IMU sensing. Therefore, we decide to give up filtering and resort again to opportunistic calibration, aiming to determine calibration opportunities when the observations do provide sufficient observability to the system. To this end, we redefine the observation o_k as the radial distance by accumulating the radial displacements, and we also revise the calculation of the residual $\mathbf{r}_k$ as:

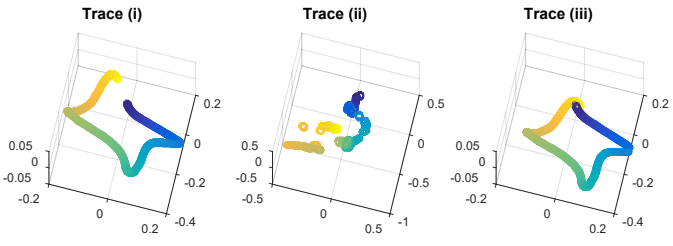
$$\mathbf{r}_k = \begin{bmatrix} r_k^x \\ r_k^y \\ r_k^z \\ r_k^{yz} \\ r_k^{xz} \\ r_k^{xy} \\ r_k^{xyz} \end{bmatrix} = \begin{bmatrix} |o_k - |p_k^x|| \\ |o_k - |p_k^y|| \\ |o_k - |p_k^z|| \\ |o_k - \|[p_k^y, p_k^z]\|_2| \\ |o_k - \|[p_k^x, p_k^z]\|_2| \\ |o_k - \|[p_k^x, p_k^y]\|_2| \\ |o_k - \|\mathbf{p}_k\|_2| \end{bmatrix}, \tag{4}$$

where we compute scalar residuals with respect to vectors derived from 7 combinations of the 3 coordinates of $\mathbf{p}_k$. Note that $\mathbf{r}_k$ in (2) is r_k^{xyz} in our calculation.

We keep using the prediction step in (2) to obtain $\hat{\mathbf{s}}_{k|k-1}$ whenever new IMU readings become available. Upon obtaining a new observation o_k, we compute $\mathbf{r}_k$ as (4), and correct $\hat{\mathbf{s}}_{k|k-1}$ using **Algorithm 1**. To facilitate the algorithm presentation, we let $[p_k^1, p_k^2, p_k^3] = [p_k^x, p_k^y, p_k^z]$; we denote by $\mathbf{p}_k^{-j}$ the position vector with the j-th component of $\mathbf{p}_k$ set to zero and by $\mathbf{p}_k^j$ the complement of $\mathbf{p}_k^{-j}$ (i.e., the j-th component of $\mathbf{p}_k$ remains while others set to zero). Essentially,



(a) Motion tracking with different tracking schemes.



(b) Corresponding motion traces.

Fig. 6. Comparison among IMU-only tracking, Kalman filter, and our opportunistic calibration. Our method preserves the motion trend while substantially reducing the drift.

Algorithm 1: Opportunistic State Calibration

Data: $\hat{\mathbf{s}}_{k|k-1}$, $\mathbf{r}_k$, t_{prv}, ϵ
Result: $\hat{\mathbf{s}}_k$, t_{prv}
begin
  $t \leftarrow \text{clock\_time} - t_{\text{prv}}$; $\hat{\mathbf{s}}_k \leftarrow \hat{\mathbf{s}}_{k|k-1}$;
  if $o_k < \epsilon$ **then**
    $\hat{\mathbf{v}}_k \leftarrow \hat{\mathbf{v}}_{k|k-1} - \hat{\mathbf{p}}_{k|k-1}/t$; $\hat{\mathbf{p}}_k \leftarrow \mathbf{0}$;
  else
    $j \leftarrow \arg\min_i [\mathbf{r}_k]_i$;
    if $r_k^j < \epsilon$ ***and*** $j \in \{1,2,3\}$ **then**
      $\hat{\mathbf{v}}_k \leftarrow \hat{\mathbf{v}}_{k|k-1} - \hat{\mathbf{p}}_{k|k-1}^{-j}/t$; $\hat{\mathbf{p}}_k \leftarrow \mathbf{p}_{k|k-1}^{j}$;
    else if $r_k^j < \epsilon$ ***and*** $j \in \{4,5,6\}$ **then**
      $\hat{\mathbf{v}}_k \leftarrow \hat{\mathbf{v}}_{k|k-1} - \hat{\mathbf{p}}_{k|k-1}^{(j-3)}/t$; $\hat{\mathbf{p}}_k \leftarrow \hat{\mathbf{p}}_{k|k-1}^{-(j-3)}$;
  $t_{\text{prv}} \leftarrow \text{clock\_time}$;

Algorithm 1 looks for three types of opportunities where o_k may offer sufficient observability: i) o_k is close to zero, ii) the magnitude of either of the three coordinates of $\mathbf{p}_k$ is close to o_k, and iii) the Euclidean norm of combining either of the two coordinates of $\mathbf{p}_k$ is close to o_k. While case i) suggests that $\mathbf{p}_k$ is caused by drifts, the other two cases indicate certain coordinate(s) (other than those concerned by the criteria) of $\mathbf{p}_k$ is(are) caused by drifts. The algorithm acts similarly under each of these opportunities: it first corrects the velocity vector by subtracting the mean velocity drift, which is computed by dividing the elapsed time t since the last calibration into the position drift. Then it sets the drift coordinate(s) of $\mathbf{p}_k$ to zero. Fig. 6 shows that, while the IMU-only tracking may drift away when the smartwatch moves according to a square shape, the "brutal" correction applied by Kalman filtering actually exacerbates the situation: it cannot even preserve the trend. Fortunately, applying our opportunistic calibration may largely recover the motion trace.

V. System Architecture

We hereby assemble the aforementioned components to construct SoM, whose architecture is illustrated in Fig. 7. The smartphone takes several duties, including constantly transmitting the tone sequence during the whole operation period and relaying the tracking results from the smartwatch to whatever devices that require wrist tracking. Using a Bluetooth/WiFi relay allows SoM to have an extended range beyond what Bluetooth may reliably reach while conserving energy for the watch. The device demanding the uHMI uses WiFi to receive the tracking states and it needs a driver to "translate" the states to what an application may require (e.g., the motion of a role in a video game). Our prototype only implements a basic driver displaying the wrist motion trace.

The SoM system core resides in the smartwatch. On one hand, the tone-driven radial tracking accumulates differential measurements to derive observations for the state calibration. On the other hand, IMU-sensing drives the independent orientation tracking, enabling SoM to adjust the accelerometer readings into wrist motion accelerations under a global coor-

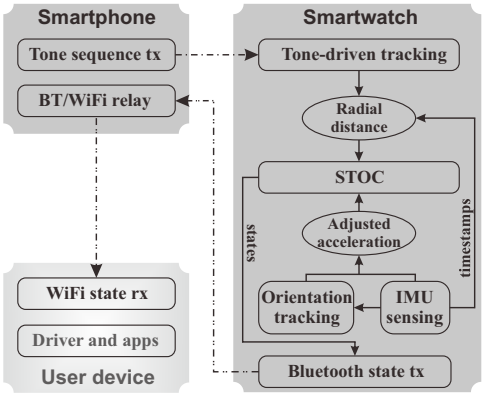


Fig. 7. System architecture of SoM.

dinate system with the smartphone as the origin. Based on the adjusted accelerometer readings and the observations, the State Transition & Opportunistic Calibration (STOC) module keeps track of the wrist positions and the results are transmitted over Bluetooth to the phone. One component not explicitly discussed is detecting *facing direction*: it is included under the orientation tracking as a sub-module, and it requires a user to initialize SoM by aligning the y-axis of the watch with the facing direction (e.g., stretching the arm straightly ahead).

Arranging all the computational components into a resource-constrained smartwatch is far from trivial. In Fig. 8, we illustrate the execution pipeline to demonstrate how to coordinate multiple threads to complete the complicated sensing/calibrating process in real-time. The Android system has two default threads for recording sounds and retrieving IMU sensor readings: while the former is done regularly at 44.1kHz, the latter yields timestamped samples arriving irregularly. We set the sound recording buffer to have a size of $l_{\text{buff}} = 512$, and the SoM ranging thread periodically takes samples from the buffer for tone detection. Upon a detection at t_{detect}, the actual tone arrival time t_{arrival} is deduced by counting the samples. The SoM STOC thread then retrieves all the sensor readings between the previous t_{arrival} and the current one;

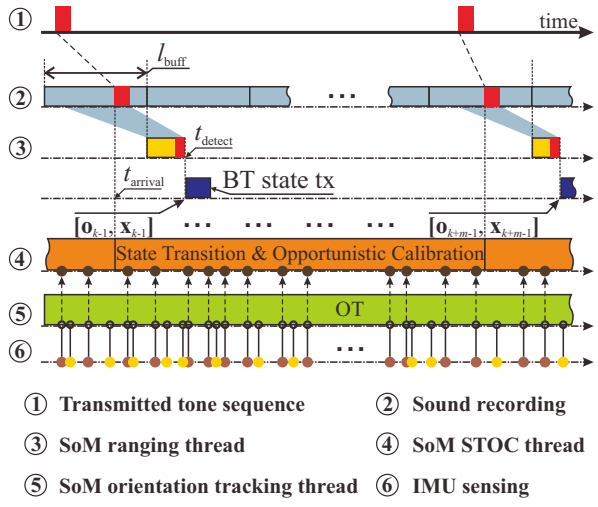


Fig. 8. Multi-threaded execution pipeline.

it aligns them with the observations, filters the accelerations, and estimates the current state according to the algorithms discussed in Sec. IV-C. The resulting state is then sent to the smartphone using Bluetooth. The orientation tracking thread runs in parallel based on all IMU sensor readings, and its output is used for adjusting accelerations.

VI. Performance Evaluation

We report extensive experiments on SoM in this section. We first explain implementation details and how the evaluations are conducted, then we present the evaluation results for SoM, including a comparison with ArmTrak [8]. We would refrain from comparing with AAMouse [7] and CAT [9] as the results are expectable: they are bounded to excel in accuracy, but SoM is lightweight and infrastructure-free.

A. Implementation Details

We have implemented the entire SoM system on a watch-phone pair. We choose LG W100 smartwatch and HTC One M8 smartphone, as watches such as Samsung Gear Live used previously [8] are not available anymore. Although we prototype the system in Android wear system (which most of the smartwatches can support), we believe the prototype can be readily migrated to other systems, given its light demand on both software and hardware. We keep the phone transmitting tones with 70ms interval, with each 220Hz tone carried upon 17kHz and lasting for 1ms. The tone detection threshold is set according to (1) in Sec. III-C, where $c = 10^6$, $\alpha = 0.5$, and $\beta = 2$. The threshold ϵ used in **Algorithm 1** is set to 0.01m. We set the IMU sensing rate to the highest level, so the sample frequency is roughly 200Hz for all IMU sensors, though their samples may still arrive irregularly.

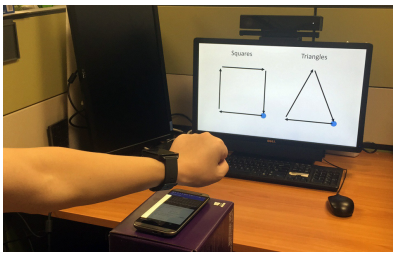


Fig. 9. Experiment setup.

We recruit 6 users (3 males and 3 females) for testing SoM. Participants can wear the watch on either the left or right wrist based on their preference since our algorithm is insensitive to user's wearing habit. They are then asked to perform a set of motion traces shown on a screen (Fig. 9), including straight lines, circles, squares, triangles and numbers (from 1 to 4) at different scales. Although example traces are on a 2D plane, we do not restrict a user's motion to any fixed plane; it can be arbitrarily in 3D space. Since our algorithm requires calibration opportunities with zero observations (i.e., $o_k < \epsilon$), the motion traces are designed so as to start and end at the phone position. However, when performing the gestures, they were not asked to purposely return to the exact starting position but just to follow the gesture trace to perform. It helps us to evaluate the performance of SoM in adequately capturing calibration opportunities.

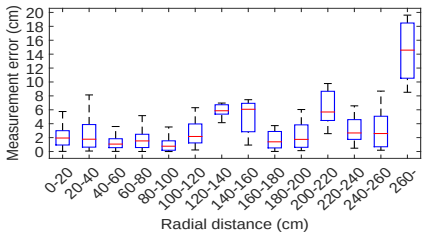


Fig. 10. Ranging errors at different radial distances.

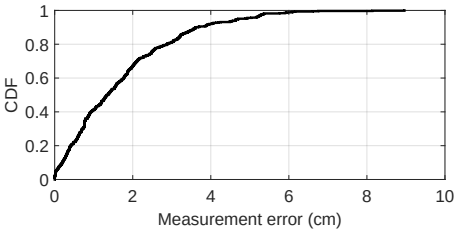


Fig. 11. Well-controlled errors at shorter distances.

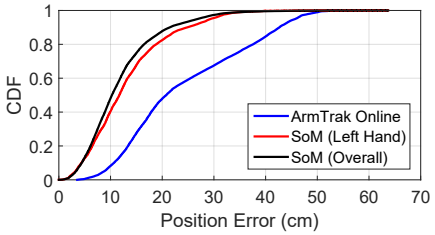


Fig. 12. Comparing SoM with ArmTrak in CDF.

To evaluate the tracking accuracy, we need certain ground truth to compare with. To make a fair comparison with ArmTrak [8], we use Kinect 2.0 and its Body Skeleton API to retrieve the estimated position of the wrist, though computer vision based tracking may cause non-negligible errors. The wrist motion traces are recorded and then analyzed offline. Since Kinect has a much lower sampling rate than SoM, we reconstruct the ground truth trace by interpolating the trace based on the time frame. The possible discrepancy incurred by the asynchrony between Kinect and SoM is controlled within 44 ms, the average sampling interval for Kinect program.

B. Acoustic Ranging Evaluation

We first evaluate the accuracy of acoustic ranging. We move the watch away from the phone, which is marked as zero, along a scale for multiple times to different distance then back to the phone, and record the ranging result in the watch for offline analysis, and the intermediate ranging distances are assumed to be linearly interpolated. We classify the errors according to their respective ground truth radial distances (Fig. 10): they are well-controlled under shorter distances; the error CDF within 1m range (Fig. 11) shows a median error below 2cm. The errors should have been monotonically increasing beyond 1m distance, but our adaptive threshold manages to keep them under control within 2m distance. The non-monotonic trend of errors beyond 1m is an artifact caused by the specific parameters chosen for the adaptive threshold: the parameters are chosen to maintain the overall performance rather than to shape the trend. In general, we would expect a uHMI to require wrist motions mild enough so that the acoustic ranging could operate in its safe zone.

C. Comparison with ArmTrak

To evaluate the overall performance of SoM, we first compare SoM with ArmTrak's online algorithm (hereafter ArmTrak). ArmTrak computes the elbow position as weighted average position of an orientation-to-position dictionary with a prior point cloud sampled by Kinect as weights. We obtain the prior point cloud by performing all the designed gestures, and the point cloud is created with a 1cm resolution. After obtaining the orientation calibrated with the facing direction, we further get the estimated elbow position then transform it to wrist position. Fig. 12 shows the CDF of wrist position error. The median error of SoM is about 11.7cm for the 2 left-hand users but is only 10.3cm overall, compared to 20.7cm for ArmTrak. Moreover, SoM manages to keep up to 90% of errors within 20cm, though its maximum error is over 60cm, slightly larger than ArmTrak's 50cm.

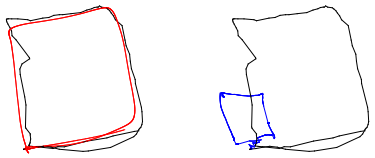


Fig. 13. Smaller scale of ArmTrak result: SoM (Red) and ArmTrak Online (Blue) vs. Kinect tracking result (Black).

We could not achieve the claimed median error in [8] probably due to the different experiment setup. As ArmTrak estimate wrist position via elbow, it is insensitive to certain motions (such as squares and lines), rendering the shape scale at wrist much smaller than the true size, as demonstrated in Fig. 13; the errors larger than 30cm occur mostly due to the scale difference. Moreover, ArmTrak assumes fixed shoulder, hereby any body movement may potentially affect its performance. For SoM, most errors result from the drift, but our STOC module can keep most of the drift largely checked. However, since ArmTrak is based on a prior point cloud, the error is well bounded by the motion scale and prior, while SoM may suffer from larger maximum error due to some uncorrectable drift at some farther distance.

Fig. 17 shows some example traces produced by SoM and ArmTrak projected on the global XZ plane. Compared with ArmTrak, SoM can achieve smoother trace and more reasonable scale since its tracking methodology follows physical laws. Moreover, SoM is not constrained by the assumption of fixed shoulder or any user's wearing behavior. SoM can also track elbow if we reversely use the rationale proposed in ArmTrak, but we will not evaluate the tracking accuracy for elbow since it is not the main point of this paper.

D. Impact of Motion Scale

Since the tracking of SoM is calibrated by the sound ranging results as shown in Sec. VI-B, the scale of wrist motion will affect the tracking accuracy. We evaluate the tracking performance across different scale of motion traces. For the ground truth of each motion trace, we define a bounding box which contains the entire motion trace, and use the diagonal distance as the scale of the motion. We categorize the scales into 3 types: small ($\leq 30cm$), middle ($30-60cm$), and large ($\geq 60cm$). Fig. 14 shows the mean and median error for motion at different scales. Both quantities are slightly increasing with the scale from 10cm to 14cm due to the error introduced in sound ranging that in turn leads to fewer chances

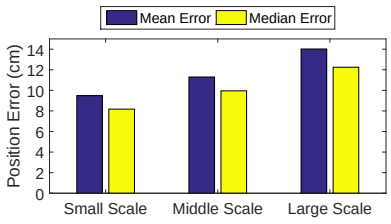


Fig. 14. Mean and median errors vs motion scales.

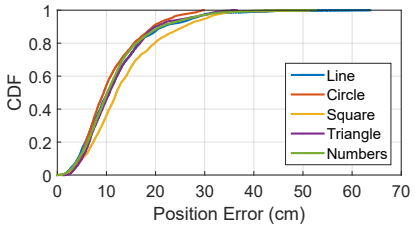


Fig. 15. CDF of error vs motion types.

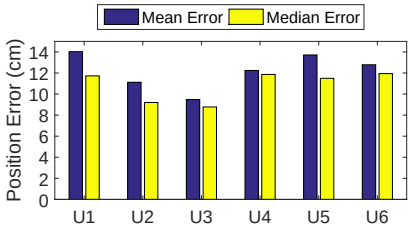


Fig. 16. Mean and median errors for different users.

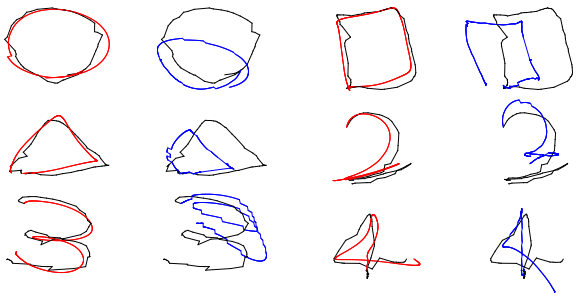


Fig. 17. Example Traces: SoM (Red) and ArmTrak Online (Blue) vs. Kinect tracking result (Black).

for opportunistic calibration. For human arm, most of gentle motion performed in a uHMI scenarios are within the scale of 60cm, for which SoM can keep the median error within 10cm.

E. Accuracy across Different Motions

We evaluate the performance of SoM across different motions. Fig. 15 plots the CDF of errors for 5 sets of motions: line, circle, square, triangle and numbers. We observe that the performance is consistent across different motions, showing that SoM is not biased to any patterns. Among all motions, circle has the best performance whereas square leads to the worst. Due to its non-smooth nature, drawing squares involves more sudden changes in moving direction compared to circle, and such a sudden switch from one axis to another may introduce errors during opportunistic state calibration. Line has the largest maximum error compared to the rest since all users tend to draw lines in a larger scale. Drawing circle has the smallest maximum error as a result of a smaller and stabler shape size when the participants performed such motion.

F. Accuracy across Users

We evaluate SoM performance across different users to test its robustness against different user behaviour. Although 6 users used different hand (U4 and U5 chose left hand, while the rest chose right hand) and moved their hand with different speed and scale, we observe that the median error across different users are all within 12cm. Among all users, U1 has the worst performance while U3 has the best performance. From the detailed tracking result, we find U1 performed the gesture in a larger scale, which are all beyond 60cm, while U3 has relatively small motion scale, which are all within 40cm. There is no obvious difference between left-hand users and right-hand users. Regarding the moving speed, we observe that both U2 and U6 performed relatively faster compared to the rest, but the performance under a higher speed is still stable.

G. Energy Consumption

Lacking dedicated devices, we cannot measure the exact energy consumption for each component in SoM. Instead, we conduct a simple experiment to roughly estimate the overall energy consumption of SoM over a long period of time. We kept both phone and watch running and tracking for 20 minutes. While tracking, we keep the watch sending data to phone every 100ms through Bluetooth, and keep the phone processing received data and sending them to PC through web socket. For both devices, we have recorded the timestamps (in seconds) for each 1% decrease in the battery level. Based on the battery capacity of HTC One M8 smartphone (2600mAh) and LG W100 smartwatch (400mAh) and the percentage and time recorded, we estimated the power consumption of the entire system over time as shown in Fig. 18. We observed that the average energy consumed per minute is about 5mAh for phone and 2mAh for watch. This consumption includes that of WiFi and Bluetooth since the communication among watch, phone and PC is a part of the SoM system as a uHMI.

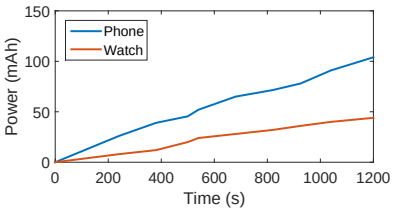


Fig. 18. Energy consumption of SoM.

VII. Conclusions

In this paper, we have presented SoM as a wrist tracking system with a lightweight design but a satisfactory tracking accuracy. SoM adopts a pair of smart watch and phone as its hardware basis, which makes it ubiquitously deployable. Using the smartphone to send sound tones, SoM derives the distance between the watch and phone. It leverages this information to opportunistically calibrate drifts caused by IMU tracking. In order to realize these functions, SoM represents a delicately engineered sensing and signal processing pipeline that fits into the resource-scarce smartwatch. Overall, SoM achieves a promising tracking performance with a median error of 10.27cm. When comparing with ArmTrak [8], we have identified certain complementary features between ArmTrak and SoM. Therefore, we plan to improve the performance of SoM by integrating the prior-driven position estimation method proposed by ArmTrak for strengthen both the intitial and opportunistic calibrations. This may lead to a system that retains SoM's detail preserving ability while having an improved robustness against drift.

REFERENCES

[1] Microsoft, "Kinect for Windows," https://developer.microsoft.com/en-us/windows/kinect.

[2] Nintendo, "Wii U," http://www.nintendo.com/wiiu.

[3] "Leap Motion," https://www.leapmotion.com/.

[4] "Binary Fortress VoiceBot," https://www.voicebot.net/.

[5] "Oculus Rift," https://www.oculus.com/en-us/rift/.

[6] "Microsoft HoloLens," https://www.microsoft.com/microsoft-hololens/.

[7] S. Yun, Y.-C. Chen, and L. Qiu, "Turning a Mobile Device into a Mouse in the Air," in *Proc. of the 13th ACM MobiSys*, 2015, pp. 15–29.

[8] S. Shen, H. Wang, and R. R. Choudhury, "I Am a Smartwatch and I Can Track My User's Arm," in *Proc. of the 14th ACM MobiSys*, 2016, pp. 85–96.

[9] W. Mao, J. He, and L. Qiu, "CAT: High-Precision Acoustic Motion Tracking," in *Proc. of the 22nd ACM MobiCom*, 2016, pp. 69–81.

[10] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *ASME Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[11] P. Del Moral, "Nonlinear Filtering: Interacting Particle Solution," *Markov Processes and Related Fields*, vol. 2, no. 4, pp. 555–580, 1996.

[12] E. Jacobsen and R. Lyons, "The Sliding DFT," *IEEE Signal Processing Magazine*, vol. 20, no. 2, pp. 74–80, 2003.

[13] Y. Yang and D. Ramanan, "Articulated Pose Estimation with Flexible Mixtures-of-parts," in *Proc. of the 24th IEEE CVPR*, 2011, pp. 1385–1392.

[14] A. Toshev and C. Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks," in *Proc. of the 27th IEEE CVPR*, 2014, pp. 1653–1660.

[15] X. Peng, L. Wang, and X. W. Y. Qiao, "Bag of Visual Words and Fusion Methods for Action Recognition: Comprehensive Study and Good Practice," *Elsevier Computer Vision and Image Understanding*, vol. 150, pp. 109–125, 2016.

[16] J. Xiong and J. Kyle, "ArrayTrack: A Fine-Grained Indoor Location System," in *Proc. of the 10th USENIX NSDI*, 2013, pp. 71–84.

[17] S. Sen, J. Lee, K. Kim, and P. Congdon, "Avoiding Multipath to Revive Inbuilding WiFi Localization," in *Proc. of the 11th ACM MobiSys*, 2013, pp. 249–262.

[18] L. Sun, S. Sen, D. Koutsonikolas, and K.-H. Kim, "WiDraw: Enabling Hands-Free Drawing in the Air on Commodity WiFi Devices," in *Proc. the 21st ACM MobiCom*, 2015, pp. 77–89.

[19] S. Yun, Y.-C. Chen, H. Zheng, L. Qiu, and W. Mao, "Strata: Fine-Grained Acoustic-based Device-Free Tracking," in *Proc. of the 15th ACM MobiSys*, 2017, pp. 15–28.

[20] W. Mao, M. Wang, W. Sun, L. Qiu, S. Pradhan, and Y.-C. Chen, "RNN-Based Room Scale Hand Motion Tracking," in *Proc. of the 25th ACM MobiCom*, 2019.

[21] Y. Wang, J. Shen, and Y. Zheng, "Push the Limit of Acoustic Gesture Recognition," in *Proc. of the 39th IEEE INFOCOM*, 2020.

[22] S. Agrawal, I. Constandache, S. Gaonkar, R. Roy Choudhury, K. Caves, and F. DeRuyter, "Using Mobile Phones to Write in Air," in *Proc. of the 9th ACM MobiSys*, 2011, pp. 15–28.

[23] T. Park, J. Lee, I. Hwang, C. Yoo, L. Nachman, and J. Song, "E-Gesture: A Collaborative Architecture for Energy-Efficient Gesture Recognition with Hand-worn Sensor and Mobile Devices," in *Proc. of the 9th ACM SenSys*, 2011, pp. 260–273.

[24] S. Gupta, D. Morris, S. Patel, and D. Tan, "SoundWave: Using the Doppler Effect to Sense Gestures," in *Proc. of the 30th ACM CHI*, 2012, pp. 1911–1914.

[25] Z. Sun, A. Purohit, R. Bose, and P. Zhang, "Spartacus: Spatially-aware Interaction for Mobile Devices Through Energy-efficient Audio Sensing," in *Proc. of the 11th ACM MobiSys*, 2013, pp. 263–276.

[26] Q. Pu, S. Gupta, S. Gollakota, and S. Patel, "Whole-Home Gesture Recognition Using Wireless Signals," in *Proc. of the 19th ACM MobiCom*, 2013, pp. 27–38.

[27] W. Wang, A. X. Liu, M. Shahzad, K. Ling, and S. Lu, "Understanding and Modeling of WiFi Signal Based Human Activity Recognition," in *Proc. of the 21st ACM MobiCom*, 2015, pp. 65–76.

[28] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, "Tracking Moving Devices with the Cricket Location System," in *Proc. of the 2nd ACM MobiSys*, 2004, pp. 190–202.

[29] K. Liu, X. Liu, and X. Li, "Guoguo: Enabling Fine-Grained Indoor Localization via Smartphone," in *Proc. of the 11st ACM MobiSys*, 2013, pp. 235–248.

[30] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan, "BeepBeep: A High Accuracy Acoustic Ranging System Using COTS Mobile Devices," in *Proc. of the 5th ACM SenSys*, 2007, pp. 1–14.

[31] Z. Zhang, D. Chu, X. Chen, and T. Moscibroda, "SwordFight: Enabling a New Class of Phone-to-Phone Action Games on Commodity Phones," in *Proc. of the 10th ACM MobiSys*, 2012, pp. 1–14.

[32] A. Wang and S. Gollakota, "MilliSonic: Pushing the Limits of Acoustic Motion Tracking," in *Proc. of the 31st ACM CHI*, 2019.

[33] Q. Lin, Z. An, and L. Yang, "Rebooting Ultrasonic Positioning Systems for Ultrasound-Incapable Smart Devices," in *Proc. of the 25th ACM MobiCom*, 2019.

[34] S. Shen, N. Michael, and V. Kumar, "Autonomous Multi-Floor Indoor Navigation with a Computationally Constrained MAV," in *Proc. of the 28th IEEE ICRA*, 2011, pp. 20–25.

[35] S. Grzonka, G. Grisetti, and W. Burgard, "A Fully Autonomous Indoor Quadrotor," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 90–100, 2012.

[36] A. I. Mourikis and S. I. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," in *Proc. of the 24th IEEE ICRA*, 2007, pp. 3565–3572.

[37] A. Martinelli, "Vision and IMU Data Fusion: Closed-Form Solutions for Attitude, Speed, Absolute Scale, and Bias Determination," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 44–60, 2012.

[38] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: Zero-effort Crowdsourcing for Indoor Localization," in *Proc. of the 18th ACM MobiCom*, 2012, pp. 293–304.

[39] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A Reliable and Accurate Indoor Localization Method Using Phone Inertial Sensors," in *Proc. of the 14th ACM UbiComp*, 2012, pp. 421–430.

[40] S. Yoon, K. Lee, and I. Rhee, "FM-based Indoor Localization via Automatic Fingerprint DB Construction and Matching," in *Proc. of the 11th ACM MobiSys*, 2013, pp. 207–220.

[41] Y. Shu, C. Bo, G. Shen, C. Zhao, L. Li, and F. Zhao, "Magicol: Indoor Localization Using Pervasive Magnetic Field and Opportunistic WiFi Sensing," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 7, pp. 1443–1457, 2015.

[42] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye, "Push the Limit of WiFi Based Localization for Smartphones," in *Proc. of the 18th ACM MobiCom*, 2012, pp. 305–316.

[43] M. Elgendi, F. Picon, N. Magnenat-Thalmann, and D. Abbott, "Arm Movement Speed Assessment via a Kinect Camera: A Preliminary Study in Healthy Subjects," *BioMedical Engineering OnLine*, vol. 13, no. 88, pp. 1–14, 2014.

[44] R. E. Kalman, "Mathematical Description of Linear Dynamical Systems," *SIAM J. Control A*, vol. 1, no. 2, pp. 152–192, 1963.

[45] H. J. Luinge., "Inertial Sensing of Human Movements," Ph.D. dissertation, University of Twente, 2002.

[46] P. Zhou, M. Li, and G. Shen, "Use It Free: Instantly Knowing Your Phone Attitude," in *Proc. of the 20th ACM MobiCom*, 2014, pp. 605–616.