SyncWISE: Window Induced Shift Estimation for Synchronization of Video and Accelerometry from Wearable Sensors

YUN C. ZHANG*†, Georgia Institute of Technology, United States
SHIBO ZHANG*, Northwestern University, United States
MIAO LIU, Georgia Institute of Technology, United States
ELYSE DALY, Northwestern University, United States
SAMUEL BATTALIO, Northwestern University, United States
SANTOSH KUMAR, University of Memphis, United States
BONNIE SPRING, Northwestern University, United States
JAMES M. REHG, Georgia Institute of Technology, United States
NABIL ALSHURAFA, Northwestern University, United States

The development and validation of computational models to detect daily human behaviors (e.g., eating, smoking, brushing) using wearable devices requires labeled data collected from the natural field environment, with tight time synchronization of the micro-behaviors (e.g., start/end times of hand-to-mouth gestures during a smoking puff or an eating gesture) and the associated labels. Video data is increasingly being used for such label collection. Unfortunately, wearable devices and video cameras with independent (and drifting) clocks make tight time synchronization challenging. To address this issue, we present the Window Induced Shift Estimation method for Synchronization (SyncWISE) approach. We demonstrate the feasibility and effectiveness of our method by synchronizing the timestamps of a wearable camera and wearable accelerometer from 163 videos representing 45.2 hours of data from 21 participants enrolled in a real-world smoking cessation study. Our approach shows significant improvement over the state-of-the-art, even in the presence of high data loss, achieving 90% synchronization accuracy given a synchronization tolerance of 700 milliseconds. Our method also achieves state-of-the-art synchronization performance on the CMU-MMAC dataset.

*Both authors contributed equally to this research.
†Corresponding Author, http://sites.google.com/view/yunzhang

Authors' addresses: Yun C. Zhang, Georgia Institute of Technology, School of Electrical and Computer Engineering and Center for Health Analytics and Informatics, Atlanta, Georgia, 30332, United States, yzhang467@gatech.edu; Shibo Zhang, shibo.zhang@northwestern.edu, Northwestern University, Department of Preventive Medicine and Department of Computer Science, Chicago, Illinois, United States; Miao Liu, Georgia Institute of Technology, School of Electrical and Computer Engineering and Center for Health Analytics and Informatics, Atlanta, Georgia, 30332, United States, mliu328@gatech.edu; Elyse Daly, Northwestern University, Department of Preventive Medicine, Chicago, Illinois, United States, elyse.daly@northwestern.edu; Samuel Battalio, Northwestern University, Department of Preventive Medicine, Chicago, Illinois, United States, samuel.battalio@northwestern.edu; Santosh Kumar, University of Memphis, Department of Computer Science, Memphis, Tennessee, United States, skumar4@memphis.edu; Bonnie Spring, Northwestern University, Department of Preventive Medicine, Chicago, Illinois, United States, bspring@northwestern.edu; James M. Rehg, Georgia Institute of Technology, School of Interactive Computing and Center for Health Analytics and Informatics, Atlanta, Georgia, 30332, United States, rehg@gatech.edu; Nabil Alshurafa, Northwestern University, Department of Preventive Medicine and Department of Computer Science, Chicago, Illinois, United States, nabil@northwestern.edu.

2474-9567/2020/9-ART107 $15.00
https://doi.org/10.1145/3411824

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**; **Ubiquitous and mobile computing design and evaluation methods**.

Additional Key Words and Phrases: Time Synchronization; Video; Accelerometry; Wearable Camera; Wearable Sensor; Temporal Drift; Automatic Synchronization

ACM Reference Format:
Yun C. Zhang, Shibo Zhang, Miao Liu, Elyse Daly, Samuel Battalio, Santosh Kumar, Bonnie Spring, James M. Rehg, and Nabil Alshurafa. 2020. SyncWISE: Window Induced Shift Estimation for Synchronization of Video and Accelerometry from Wearable Sensors. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 3, Article 107 (September 2020), 26 pages. https://doi.org/10.1145/3411824

1 INTRODUCTION

The temporally-precise annotation of sensor data is necessary in order to build models that can passively sense and infer behavior from sensor signals. For behaviors involving limb movements, such as smoking, eating, and brushing, video-recordings from wearable cameras are increasingly being used to obtain temporally-precise ground truth labels. The cameras are worn and positioned to capture movements of interest under field conditions (see Fig. 1). Video is recorded simultaneously with the target mobile sensor data, and standard video coding is used to obtain ground truth labels for the sensor streams. These data can be used both to validate the accuracy of existing methods and to train new models. This approach has been used for eating, drinking, and brushing activities [1–4, 9, 45], and is particularly valuable for fine-grained activities lasting on the order of seconds. However, this approach requires *accurate time synchronization* between the video sequence and the sensor data streams so that annotations obtained from video can be automatically transferred to label the sensor data. Any temporal misalignment between the video and sensor streams will result in label noise (i.e., incorrect labeling of the sensor data) and can significantly degrade the accuracy of the detector.

Because it is common for commodity sensor hardware to utilize independent, unsychronized clocks, previous sensor system architectures have incorporated effective approaches to sensor synchronization [12, 25, 38]. Unfortunately, these approaches cannot be easily extended to video capture. In contrast to sensor network approaches [38], commercially-available wearable video cameras such as GoPro, are not designed for synchronization with other non-camera sensors.[1] In addition, battery constraints make it infeasible to transmit video data wirelessly so that it cannot be time-stamped at a central collection point simultaneously with other sensor streams [25]. As a result of these issues, the problem of time synchronizing of video cameras [20, 37, 41] and wearable devices [35] is well-known within the community to be a practical challenge in study implementation and a silent killer of data accuracy [4].[2]

When data is collected under laboratory conditions, many strategies can be used to establish synchronization points, such as the well-known clapperboard for audio-visual (AV) synchronization or the use of special hand gestures to synchronize body-worn accelerometers with cameras [34, 35]. These approaches are impractical for field studies as they impose significant burden on participants and rely on their adherence [10]. Alternatively, manual synchronization can be performed with tools such as ELAN [11] or Chronoviz [18]. This approach is laborious and time-consuming, and as a result of clock drift, it may need to be performed at multiple time points across a long recording.[3] It follows that there is a need for a flexible, general purpose solution for synchronizing

[1] While add-on products from third-party vendors, such as SyncBac Pro and ':pulse' from Timecode Systems [30], can provide synchronization solutions for wearable cameras, they are limited to synchronizing multiple cameras and do not address our scenario.
[2] Even something as basic as synchronizing audio and video for film-making has a long history of challenges and failures. For example, a legendary live performance by Aretha Franklin in 1972 was not released for 46 years, due in part to the failure to synchronize the video and audio properly during recording [19].
[3] Several authors have investigated clock drift arising in video cameras [33, 42]. The GoPro has an average drift of 1 second per hour, which is roughly the same duration as many fine-grained gestures.

cameras with other mobile sensors that can be applied to field-collected data, does not impose any additional burden on participants, and is fully-automatic.

In this work, we introduce a fully-automatic method called Window Induced Shift Estimation for Synchronization of video and accelerometry (SyncWISE). Given a clip of video and accelerometry data, it outputs the time offset for synchronization (see Fig. 1(a)). We address the two key technical challenges of *partial observability* and *coordinate registration*. Partial observability refers to the fact that the time intervals in which synchronization points can be reliably identified are sporadically distributed. For example, a chest-worn camera on a participant standing at a street corner will capture significant dynamic video content, while a co-located accelerometer will register no movement. This is in contrast to prior work [13, 21] which has implicitly assumed that all moments of time are equally good for estimating synchronization. We address partial observability via a kernel density estimation approach in which weighted segment pairs are correlated and their votes aggregated to obtain the final offset. The second challenge of coordinate registration arises in synchronizing video with motion-based sensors, such as accelerometers, that output their data with respect to a 3D coordinate system. In this case, the correct comparison of the signals requires the two coordinate systems (camera and sensor) to be registered, so that corresponding directions of movement are being compared. In contrast, prior work on sensor synchronization in autonomous vehicles [21] leverages the fact that sensors are rigidly mounted and calibrated during installation. We address coordinate registration by using a PCA analysis to identify a common principle direction between modalities prior to registration. We validate our SyncWISE method on two datasets: the CMU-MMAC activity dataset [16], and a novel real-world dataset, called Sense2StopSync *(S2S-Sync)*, from a smoking cessation field study with 21 participants, consisting of 45.2 hours of recordings over the three days prior to quit. This work makes the following three contributions:

- We introduce the *SyncWISE* method for automatically synchronizing video clips with motion-based sensor data such as accelerometers and Inertial Measurement Units (IMUs). We believe we are the first to identify and address the challenges of partial observability and coordinate registration that arise in the field environment.
- We provide the novel Sense2StopSync *(S2S-Sync)* dataset to the research community,[4] comprising 45.2 hours of time-synchronized optical-flow videos and accelerometery data from two chest-worn devices collected from 21 subjects with annotations of smoking and feeding gestures.
- We present state-of-the-art automatic synchronization results for the CMU-MMAC and S2S-Sync datasets, which significantly outperforms two versions of a baseline method [21]. The software will be made freely-available.

2 RELATED WORK

The time synchronization of multiple sensor streams is a long-standing challenge that cuts across a broad range of application domains and has a long history, ranging from the invention of the clapperboard in 1931 to synchronize audio and video during movie filming, to the protocols used to synchronize sensor networks [12]. This review is focused on methods for synchronizing video with wearable sensor streams for mobile sensing applications. We identify four categories of approaches: 1) Naturalistic methods, of which our work is an example, which do not impose any special requirement on signal capture; 2) Explicit methods, which enforce synchronization at the hardware or software level during capture; 3) Participant-based methods, which require specific actions by participants to achieve synchronization; and 4) Manual approaches which rely on human observation of video and other signals to identify synchronization points.

[4] https://github.com/HAbitsLab/SyncWISE

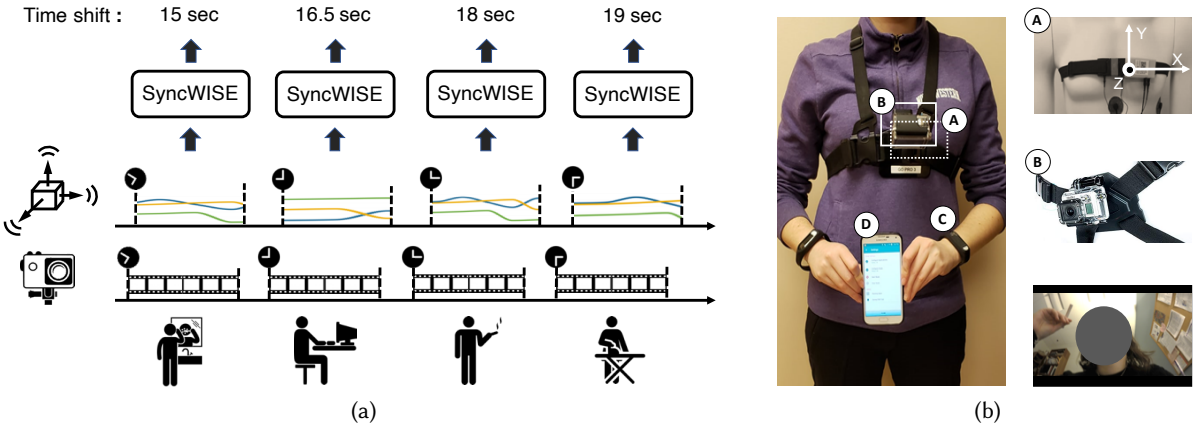


Fig. 1. (a) Illustration of input and output in our SyncWISE system. (b) The wearable sensory platform consists of (A) a chest-worn sensor suite containing a 3-axis accelerometer worn underneath the clothes, (B) a GoPro video camera; an example of video camera footage is provided below the camera, (C) a wrist-worn sensor containing a 3-axis accelerometer and a 3-axis gyroscope worn on both wrists, and (D) a study smartphone with data logging software.

2.1 Naturalistic Methods

The goal of these methods is to handle sensor data captured in the field without special hardware or specific participant behaviors. The closest previous work to ours is Fridman *et al.* [21], which describes a cross-correlation-based method designed to synchronize multi-modal signals for research in autonomous driving. Their approach assumes that all moments in time are equally good for synchronizing signals, and they use global cross-correlation to utilize the maximum amount of data. This is effective because their sensors are rigidly mounted and the coordinate axes are aligned and calibrated. In contrast, mobile wearable sensing is plagued by much greater sensor noise (due to sensors being worn improperly), variable alignment between sensor axes, and partial observability, meaning that sensors do not always capture the same phenomena with the result that not all moments in time are equally plausible for synchronization. Our matching approach, which uses windowed cross-correlation in a weighted kernel density estimation framework, addresses partial observability by identifying which windows of data provide reliable signals for synchronization. Our PCA-alignment approach provides a means to automatically align the coordinate frame axes across multiple sensors. Our experimental evaluation in Sec. 5 demonstrates the benefits of our approach over the baseline method from [21] on two datasets.

A related set of naturalistic methods provide synchronization solutions for GPS navigation systems, of which [33, 39] are representative examples. Skog *et al.* [39] provide a Kalman filter-based solution for clock drift that exploits the fact that both GPS-receiver and IMU provide signals that directly relate to the spatial location of the sensor system. In contrast, in our setting the optical flow we compute from the video cannot be directly related to the accelerometry stream due to the partial observability problem.

Another set of related methods address the automatic synchronization of multiple *video streams* [37, 41, 43]. These approaches leverage the fact that video is a single modality with unique spatiotemporal properties. In contrast, our work addresses the case of synchronizing across sensor modalities, which requires the extraction of an appropriate feature representation from each sensor's signal. Related work by Chung and Zisserman [13] uses deep learned representations to align audio and video streams in the context of correcting lipsynch effects in video dubbing. Their solution exploits the fact that the video and audio signals are always directly correlated,

unlike our case where partial observability is common. Finally, multiple prior works assume that synchronized audio-video signals are available and construct joint audio-visual feature representations for tasks such as source separation or sound classification [6, 23, 44]. While some of these works use artificial time shifts between the audio and video channels as a means of data augmentation, they have not been utilized for signal synchronization.

2.2 Explicit Synchronization

A wide range of signal capture solutions have been designed which enforce synchronization at the hardware-software level. Here we focus on three approaches. The first approach is used in sensor networks [12, 28, 38], including body area networks. Since all sensors are on the same network, protocols can be used to keep the sensor clocks synchronized, and corrections can be applied to address clock drift or skew [12]. In the second approach, all sensor signals can be wirelessly transmitted to a centralized collection node, such as a smartphone, where they are time-stamped to a common clock, thereby achieving synchronization (mCerebrum [25] is a representative example). These two approaches do not work for wearable cameras, due to lack of network support, network bandwidth, and battery limitations. An exception is when all data collection takes place in the same location. In [1], a smartphone holder is installed in the location where tooth-brushing occurs, allowing video to be recorded on the smartphone camera (the centralized node) itself, thereby achieving synchronization. In [16], cooking activities were captured in the lab, enabling a wearable camera and other sensors to be synchronized via hardware (e.g., using genlock where a reference signal from one device is used to synchronize all other devices). Note that we use the dataset from [16] for the experiments in Sec. 5.4. The third approach uses special hardware to achieve real-time synchronization [7, 14]. In [14], a periodically blinking LED is controlled to provide cues to synchronize different modalities. While such an approach can be effective, it requires additional implementation and system complexity, and the automated detection of the LED signal may be challenging in uncontrolled environments. Our approach leverages commodity hardware and standardized research grade mHealth solutions to support a broad range of study designs.

2.3 Participant-Based Methods

The clapperboard approach to audio-visual synchronization provides a reliable solution because it introduces an explicit synchronization point which is visible across modalities. Analogous approaches exist for other multi-sensor synchronization tasks. In Plötz *et al.* [34], specific hand gestures are assigned to participants to provide explicit synchronization points for aligning video and accelerometer data. Similarly, Han *et al.* [22] propose a method to synchronize video and sensor data for walking behaviors by detecting and matching the maximum backward swings of the leg. Bannach *et al.* [8] develop a method to automatically detect specific gestures (e.g., 'clap') assigned to participants. These approaches can work in controlled settings, but they introduce additional participant burden and a single point of failure in the mobile setting.

2.4 Manual Synchronization

In cases where alternative synchronization approaches fail, a fall-back solution is to use tools such as ELAN [11] or Chronoviz [18] that enable the manual identification of synchronization points via inspection. This approach has been used routinely in prior mHealth and mobile sensing works and should be considered the default method [2, 3, 27, 35, 45]. Our goal is to remove the need for such manual efforts and provide a fully-automatic solution to this important practical problem.

3 STUDY DESIGN AND DATA COLLECTION

We now detail the collection of the S2S-Sync dataset. We first describe the study design in Sec. 3.1. The sensor data collection is outlined in Sec. 3.2 and the approach to data annotation is described in Sec. 3.3.

3.1 Study Design

Data was collected during a smoking cessation study, *Sense2Stop*. Participants (age 18-65) were eligible for Sense2Stop if they had smoked at least 1 cigarette per day for the past year. The S2S-Sync dataset is generated from Sense2Stop during the three-day pre-quit period in which subjects exhibited maintenance behavior (i.e. typical smoking patterns). The pre-quit period provided baseline data for participants' smoking and eating behaviors. The video collected during pre-quit supports the annotation of smoking and eating behaviors to validate and refine machine-learned models.

3.1.1 Study Timeline. On Day 1, participants visited the lab, where they were fitted with the mobile devices and received instructions. The pre-quit phase ended on Day 4, when participants returned to the lab to upload their wearable video data to the study servers. During this visit, participants had the opportunity to delete any video footage that they did not wish to share. Participants then continued into the post-quit period without the video camera.

3.1.2 Participant Instructions. During the pre-quit period, participants were instructed to wear the provided GoPro camera for 4 hours on at-least 2 separate days, that included at least one smoking event and the eating of at least one meal and one snack, for a total of 8 hours of in-the-wild recorded video.

3.2 Devices

The wearable devices worn by the participants included a GoPro video camera strapped to their chest, a chest-worn sensor suite comprising an accelerometer, electrocardiography (ECG) sensor and respiratory plethysmography (RIP) sensor, and a pair of wrist-worn devices with tri-axial accelerometers and gyroscopes on each wrist. Additionally, they were provided with a study-dedicated smartphone with data collection software installed. We focus our analysis on synchronizing between the chest-worn accelerometer and GoPro video camera.

3.2.1 Video Camera. Participants wore a GoPro Hero 4 camera recording 1080p video at 30 Hz. The GoPro was mounted on the chest using a chest mount strap and case that protects the camera and image quality from dust, water, and other elements. The camera was oriented towards the participant's face. The captured video was stored on an µSD card as a series of MP4 files, each of which is 4 GB and 17 minutes and 43 seconds long. Before deployment, the GoPro's clock was synchronized with a PC to the National Institute of Standards and Technology (NIST) time server. As an added precaution, the camera is oriented towards the PC to briefly record the NIST time webpage (time.gov), providing an additional synch reference before the camera goes out into the field.

3.2.2 Accelerometers and Data Logging. There are two sets of accelerometers used in our study. The accelerometer from the chest-worn device, AutoSense [17], sampled at 10.66 Hz, was used in all of our automatic synchronization experiments. Note that this device is mounted on a harness which is separate from the GoPro. In Fig. 1(b), the accelerometer is on (A) while the camera is on (B). Thus while the two devices are roughly co-located, the camera is capable of significant movement relative to the accelerometer, including changes to its orientation. Additional accelerometers from the MotionSense wristband [32], mounted one on each wrist and sampled at 16 Hz, were used by the annotators during manual synchronization (see Sec. 3.3.1).[5]

Data from all accelerometers was transmitted to the study phone wirelessly and logged on an encrypted µSD card by the open-source mCerebrum smartphone app [25]. Then the data were periodically uploaded to a secure server running the open-source Cerebral Cortex [24]. The mCerebrum app time-stamps each packet of data to a common clock, thereby synchronizing the accelerometry signals to each other. Wireless transmission can result in dropped packets and packets arriving out-of-order. The software performs interpolation for small gaps in

[5]The difference in the sampling rates for the accelerometers is due to their different use cases, and the importance of sampling minimally so as to preserve battery life: The chest sensor monitors respiration, while the wrist sensors monitor physical activity.

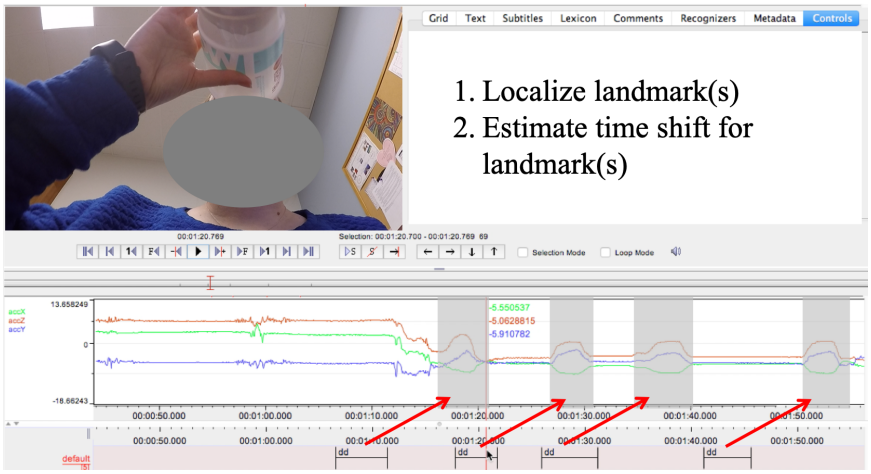


Fig. 2. Illustration of manual determination of the time shift between a wrist-worn accelerometer and chest-mounted GoPro camera for a drinking gesture, with landmarks defined using the ELAN annotation tool. Once the time-shift is identified, the label for the segmented drinking event can be transferred to the accelerometer signal.

the data, but significant gaps in the accelerometry signal remain. Our approach to synchronization explicitly accounts for missing data and poor signal quality, which is endemic to mHealth applications [26, 36].

3.3 Data Screening and Annotation

The unit of data for our experiments is a video clip with an associated segment of accelerometry data. Each clip corresponds to one MP4 file captured by the GoPro, with a maximum duration of 17 minutes and 43 seconds. Clips that were shorter than 30 seconds were discarded, resulting in 378 clips from 34 participants. The time stamps recorded by the GoPro camera are used to identify the segment of accelerometry data which is paired with the clip. This is an extremely crude correspondence with substantial error. Additional screening steps, detailed in Appendix A.1, resulted in the final S2S-Sync dataset comprising 163 video clips of 45.2 hours in total duration with associated accelerometry data from a total of 21 participants. Each participant contributed an average of 2.15 hours of usable data for analysis.

3.3.1 Ground Truth Annotation Process. Because our data capture process uses separate clocks for the video and accelerometry signals (GoPro clock and study phone clock, respectively), manual alignment of the video and accelerometry signals in each clip was performed in order to establish ground truth for our experiments. In our approach, we chose the accelerometer clock as the reference timeline, and we selected the synchronization offset that shifts the video into alignment.

The manual synchronization process, illustrated in Fig. 2, comprises two stages: a landmark video detection phase, and an accelerometer alignment phase that aligns the detected video landmarks with one of the two accelerometers (wrist- or chest-worn accelerometer). A video landmark event is defined as a distinguishable human movement (often a transition from inactivity to activity) which is visible in the video and potentially noticeable in either the wrist- or chest-worn accelerometery signals. All accelerometers are utilized in order to increase the number of available synchronization points. We find that hand movements near the face frequently result in wrist accelerometer to video matches, while transitions from sitting to standing or from standing to walking often result in chest accelerometer to video matches. After the landmark event has been localized

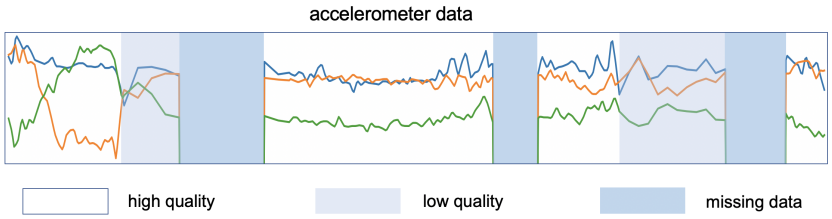


Fig. 3. A sample of 3-axis sensor signal. The accelerometer data comprises sections considered either high-quality, low-quality, or missing (no data).

in the video, the annotator will utilize a combination of cues to match sudden changes in either the wrist or chest accelerometer sensor signals to the corresponding hand or body movements in the video. After they are successfully matched, the time offset can then be calculated, and the video time is adjusted to align with the accelerometer based on this offset. The aligned signals are then inspected to determine if the estimate is sufficiently accurate. If errors remain, the process is repeated for other landmark events, resulting in additional offset measurements. The final offset is produced by combining these measurements. Fig. 2 illustrates the process of aligning a drinking event shown in the video frame with the corresponding wrist accelerometer motion using the ELAN [11] annotation tool.

3.3.2 Annotator Agreement. In order to evaluate the consistency of manual labeling, three trained annotators processed ten video clips and obtained independent estimates of the ground truth offset times. The average difference in their offset times was 346 ms, with 309 ms standard deviation. This demonstrates an unavoidable error in ground truth acquisition of fine-grained labeling and time synchronization of about 346 ms. We also conducted a one-way repeated measures ANOVA (used to determine whether three or more group means are different when the participants are the same in each group). The result illustrates no significant difference between the three annotators (F=0.60, P=0.56). Therefore, we conclude that there is no statistically significant difference between the annotations from different annotators. On average, an annotator spends 25 minutes to synchronize each video (spanning 17 minutes of data) manually. Our work aims to mitigate this problem of time synchronization, thus saving researchers thousands of hours of manual time synchronization.

4 METHODOLOGY

In this section, we introduce our approach to solving the problem of time synchronization between video and accelerometry. We start by presenting notation used, data preprocessing, and then present our time synchronization algorithm Window Induced Shift Estimation (SyncWISE), and finalize by discussing our evaluation metric.

4.1 Notation

We define $\{\mathbf{x_t} \in \mathbb{R}^2\}_{t=1}^{T}$ to be the observed motion acceleration estimated from each frame of video, and $\{\mathbf{y_t} \in \mathbb{R}^3\}_{t=1}^{T}$ to be the observed acceleration from the accelerometer in the chest-worn device. We ensure the signals are resampled to have the same frequency. Our goal is to estimate the shift $\mathbf{\Delta} \in \mathbb{R}$ between two time series such that $\mathbf{x_t}$ and $\mathbf{y_{t+\Delta}}$ describe the sensed behavior happening at the same time.

4.2 Data Preprocessing

Preprocessing has two tasks: 1) identify high-quality windows of accelerometry data to support matching, and 2) extract a motion signal from the video that can be compared to the accelerometry signal.

4.2.1 Accelerometer Data Preprocessing. As illustrated in Fig. 3, different portions of accelerometry data can vary significantly in quality due to missing data points, as the result of wireless transmission problems. To reliably screen for poor quality data, we analyze the data in one second long *segments* and define a per-segment reliability metric as the ratio of the number of data points collected in a second divided by the expected sampling rate of 10.66 Hz. We then label each segment of data as high-quality if the reliability is above 75% (i.e., 8 samples out of the 10.66 samples per second are present). Segments that do not meet this threshold are labeled as low-quality. In order to perform matching, we aggregate high-quality segments into *windows* of data that are 10 seconds long. A viable window will consist of 10 consecutive segments (i.e., 10 seconds) that were labeled high-quality. We sweep across the video clip with a one second stride to segment the viable matching windows. In order for a 17 minute video clip to be synchronizable, we require that it contain at least 10 viable windows (each 10 seconds long). Each extracted window is upsampled from 10.66 Hz to 30 Hz to match the camera sampling rate.

Note that our approach uses windows of high-quality accelerometry data as a starting point for matching to windows of video data in our SyncWISE algorithm (discussed in Sec. 4.3). This approach assumes that there are no dropped video frames, which is true for the datasets we used in our experiments. Specifically, the GoPro used in S2S-Sync stores frames locally (see Sec. 3.2.1) and the data in CMU-MMAC was collected in a lab setting. It is likely that our approach could be extended to accommodate small numbers of dropped frames, but we make no claims for efficacy in this case. Addressing large numbers of missing frames simultaneously with missing accelerometry is a topic for future work.

4.2.2 Motion Estimation from Videos. In order to compare video and accelerometry, a key operation is to extract an estimate of acceleration from video movement. This is accomplished in two steps. First, an estimate for the velocity at every pixel in every frame, known as optical flow, is computed from each pair of adjacent frames in the video. Motion features have been used in recognizing daily activities from first-person videos [29, 31, 46]. We use a deep-learning based dense optical flow estimation framework called PWC-net [40]. In the resulting 2D vector field, the vector at each pixel location provides a motion estimate of the pixel in x (horizontal) and y (vertical) directions. The second step extracts a scalar acceleration signal from the sequence of flow fields. To do this, we average the optical flow spatially for each frame and then compute the difference in the average optical flow between the current frame and the previous frame. This provides a 2D camera acceleration feature vector for each frame t, denoted as $\mathbf{x_t}$ in Section 4.1.

4.2.3 PCA Projection. An important aspect of the video data is that the orientation of the camera can be arbitrary, and this in turn affects the orientation of the acceleration vector computed in Sec. 4.2.2. While the participants were instructed to orient the camera towards their head, in practice we observed many different orientations in the dataset. The orientation can vary day-to-day as the camera harness is put on and taken off each day. This results in coordinate transformations between the sensors that vary both within-subjects across time and between-subjects. While the baseline approach [21] compares signals from a fixed pair of axes to determine the synch points, we need a way to compare axes which vary across time within the dataset.

Principle Component Analysis (PCA) allows us to map each of the 2D video data and 3D accelerometery data to a single dimension (1D) that captures the motion along the most dominant axis. To extract the 1D signal that best captures motion in a single axis, we project data from each sensing modality (both the video and accelerometer signal separately) onto their first principle component direction estimated by PCA. The first principle component corresponds to the direction of greatest variation in the data, and this is well-matched to our goal since cross-correlation (used in our proposed algorithm) leverages the variation in the signals. Formally, for video data $\{\mathbf{x_t}\}_{t=1}^{T_{V_k}}$ and accelerometer data $\{\mathbf{y_t}\}_{t=1}^{T_{V_k}}$ collected during video V_k, we have p_{vid1} and p_{acc1} as the first principle components from PCA calculated separately on $\{\mathbf{x_t}\}_{t=1}^{T_{V_k}}$ and $\{\mathbf{y_t}\}_{t=1}^{T_{V_k}}$. We then denote the 1D projected time series

as $\{\mathbf{x_t^{(p1)}} = p_{vid1}^{\mathrm{T}}\mathbf{x_t}\}_{t=1}^{T_{V_k}}$ and $\{\mathbf{y_t^{(p1)}} = p_{acc1}^{\mathrm{T}}\mathbf{y_t}\}_{t=1}^{T_{V_k}}$. We omit the subscription $p1$ for abbreviation throughout the paper.

4.3 SyncWISE Algorithm

Our approach to matching noisy video and accelerometry signals captured from mobile devices has two key components. The first is the procedure described in Sec. 4.2.1 and illustrated in Fig. 3, which selects high-quality windows of accelerometry data for matching, as a way to overcome the noise and missingness in this signal. The second key component uses weighted kernel density estimation (wKDE) to combine noisy estimates of the shift produced from multiple accelerometry-video window pairs to obtain an accurate estimate. It utilizes the cross-correlation response from each window pair to obtain a weighted Gaussian kernel and combines these kernels to estimate the offset.

The process of offset estimation is illustrated schematically in Fig. 4. The top of the figure shows the signals from a video clip and corresponding accelerometry clip separated by a ground truth offset of 1.5s. Given a window of accelerometry data, we search for the offset by shifting the video window by different amounts, to examine corresponding segments in the video. This is illustrated at the top of the figure. We show three different accelerometry windows in blue, cyan, and green. Each window will be shifted multiple times to search for potential matches. One of the corresponding window locations in the video signal is shown as a solid outline, the other locations are shown with dotted outlines. Given each pair of windows, cross-correlation (CC) is used to produce an estimate of the shift from that pair. This is illustrated in the lower part of Fig. 4 for each of the three window pairs shown with solid outlines. After detecting the peak in the CC function, a Gaussian kernel is fit to the data. Once all pairs have been correlated, a probability density function (PDF) for the global offset between the signals is constructed via a weighted sum of the Gaussian kernels (i.e., a wKDE for the offset PDF). Effectively, each window pair is casting a weighted vote for the offset. A single estimate for the offset is obtained by detecting the peak in the PDF, resulting in an estimate of 1.4s with an error of 100ms in this schematic example. The confidence score for the final estimate is obtained by fitting a Gaussian to the PDF to obtain the variance (24 in this example). Pseudocode for the method is provided in Appendix A.2. We now describe each step in detail.

4.3.1 Window Pair Sampling. Our starting point is a window w_i of accelerometry data of length T_w with samples denoted as $\mathbf{y^i} = [y_{s_i}, y_{s_i+1}, \ldots, y_{s_i+T_w}]$, where s_i is the time index of the start of window i. We generate N_s different offsets o_j^i for window i, where $j = 1, \ldots, N_s$ and $|o_j^i| <= T_{max}$. For each offset j, we obtain corresponding video samples denoted as $\mathbf{x^{i,j}} = [x_{s_i+o_j^i}, x_{s_i+o_j^i+1}, \ldots, x_{s_i+o_j^i+T_w}]$. Each o_j^i defines a window pair (see Fig. 4) for matching via cross-correlation in Sec. 4.3.2.

This approach has three design parameters: T_w which controls the window width, T_{max} which controls the search range, and N_s which controls the number of offsets. They are illustrated in Fig. 4. The consequences for these parameter settings are discussed in Sec. 4.3.4 and their optimization is discussed in Appendix A.3. Note that there are many possible ways to generate the N_s offsets. The most straight-forward approach would be to uniformly sample the search range in fixed steps. Alternatively, if a prior estimate for the shift is available (for example from knowledge of the capture setup or manual inspection), then sampling from a prior distribution could focus the window comparisons on the more likely offsets (as in importance sampling). In our experiments, we sampled offsets at random from a uniform distribution over T_{max}. We confirmed experimentally that this approach was indistinguishable from covering the search range in fixed steps.

4.3.2 Window Pair Matching. Given accelerometry-video window pairs $\mathbf{y^i}$ and $\mathbf{x^{i,j}}$, as defined in Sec. 4.3.1, we match them by calculating the cross-correlation [21, 34] to obtain an estimate $\delta_{i,j}$ for the shift between the clips.

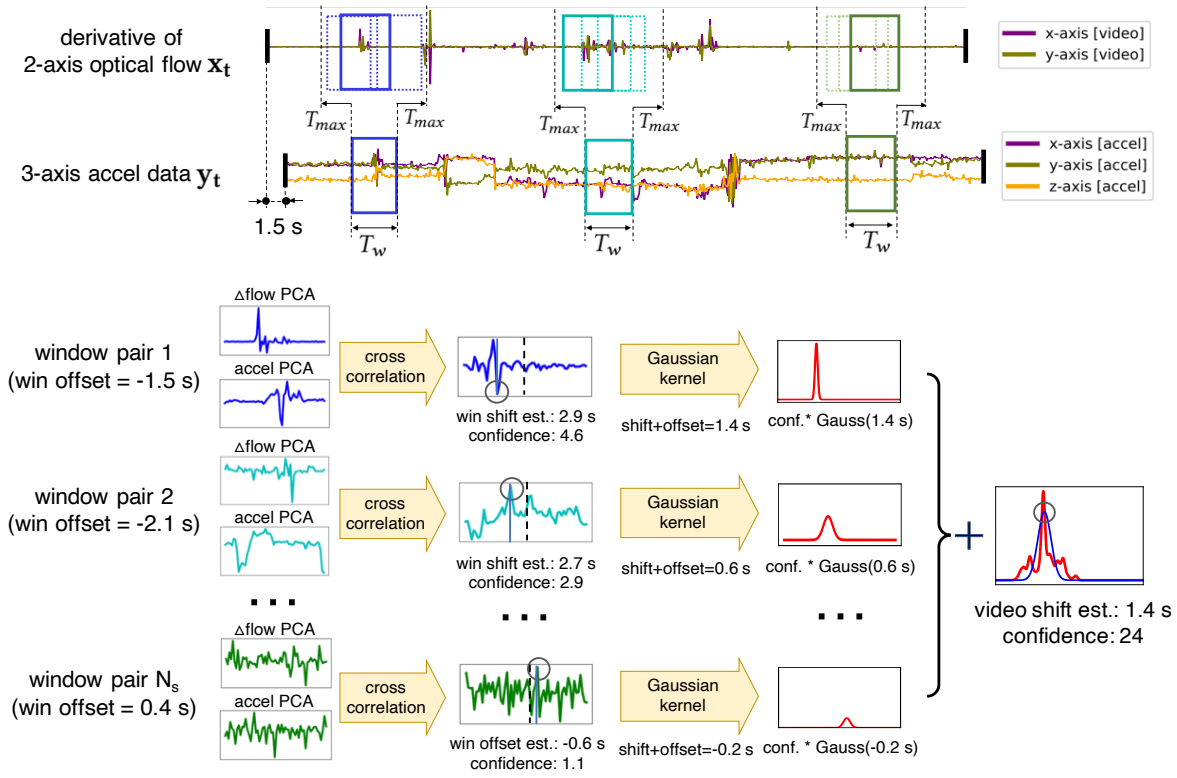


Fig. 4. SyncWISE algorithm overview: for each window that contains high quality accelerometry data, we search for the offset by shifting the video window by different amounts, to examine corresponding segments in the video (shown as blue, cyan, and green boxes). Given each pair of windows, cross-correlation (CC) is used to produce an estimate of the shift from that pair. A probability density function for the global offset between the signals is constructed via aggregating estimates from all window pairs using weighted Kernel Density Estimation as illustrated in the lower part of the figure.

Specifically, we calculate the cross-correlation function:

$$cc_{i,j}(\tau) = \sum_{n=1}^{T_w} y_n^i * x_{n+\tau}^{i,j}, \quad \tau \in [-T_w, T_w], \tag{1}$$

using an efficient fast Fourier transform (FFT)-based approach. The optimal shift for this pair of accelerometery and video data is then estimated by choosing the τ that maximizes the absolute value of the cross-correlation function centered by its median (to normalize the values) and shifted by o_j^i (its pre-generated offset):

$$\delta_{i,j} = \arg\max_{\tau} \left| cc_{i,j}(\tau) - \underset{t\in[-T_w,T_w]}{\text{median}} \{cc_{i,j}(t)\} \right| + o_j^i. \tag{2}$$

Note that each $\delta_{i,j}$ represents an estimated shift obtained from accelerometry window w_i and offset o^i_j. We then obtain a confidence score for the estimated shift as follows:

$$conf_{i,j} = \frac{\max_\tau \left| cc_{i,j}(\tau) - \underset{t\in[-T_w,T_w]}{\text{median}} \{cc_{i,j}(t)\} \right|}{\underset{t\in[-T_w,T_w]}{\text{std}} \{cc_{i,j}(\tau)\}}. \tag{3}$$

The confidence score captures the fact that not all window pairs are equally informative regarding the true offset. Intuitively, higher and sharper peaks are associated with more reliable estimates.

4.3.3 Synchronization Offset Estimation. The procedures described in Secs. 4.3.1 and 4.3.2 are repeated M times for windows of accelerometry data selected from the input clip. This results in a set of MN_s shift estimates and associated confidence values $\{\delta_{i,j}, conf_{i,j}\}$. These inputs are used to obtain a weighted kernel density estimate (wKDE) for the distribution of possible offset times between the two clips. This is constructed via a confidence-weighted voting approach as follows:

$$f(t) = \frac{1}{\sum_{i=1}^{M}\sum_{j=1}^{N_s} conf_{i,j}} \sum_{i=1}^{M}\sum_{j=1}^{N_s} conf_{i,j} \cdot K(t|\delta_{i,j},\sigma), \tag{4}$$

where $K(t|\delta,\sigma)$ is a Gaussian kernel function with mean δ and standard deviation σ. The function $f(t)$ is a pdf over the range of offsets for the input clips, which is constructed by summing weighted Gaussian kernels from the M windows and N_s shifts. The final estimated video shift Δ is attained where $f(t)$ is maximum, as follows:

$$\Delta = \arg\max_t f(t). \tag{5}$$

It can be valuable to have a confidence score associated with the estimated offset Δ. For example, given a pair of clips to be matched, the confidence score could be used to determine if the number of window samples M is sufficient. We estimate the confidence in the offset estimate by using non-linear least squares [15] to fit a Gaussian curve $g(t) = \mathcal{N}(\mu,\sigma)$ to $f(t)$, and using the variance $\hat{\sigma}$ and the variance of mean $var(\hat{\mu})$ of the gaussian to inform the confidence. The intuition here is that, if the variance of the gaussian is high, then the confidence in the delta offset is lower, and if the variance of the estimated mean is high, then the confidence is also lower. The estimated variance $\hat{\sigma}$ and the variance of the estimated mean $var(\hat{\mu})$ predicts how well $f(t)$ fits a Gaussian distribution. We define the confidence score as follows:

$$C = \frac{Const}{\hat{\sigma}\cdot var(\hat{\mu})}. \tag{6}$$

In this work, *Const* is empirically set to be 200,000. Appendix A.4.2 gives some examples of this process.

4.3.4 Discussion of Impact of Parameter Choices. The method depends on the choice of window size T_w, maximum offset T_{max}, and number of window pairs (shifts) N_s. Only windows sampled near the ground truth offset can contribute positive votes to the response curve $f(t)$. The percentage of positive votes is at most T_w/T_{max}. So larger T_w and smaller T_{max} values are preferred. However, the effect of T_w also depends on the data quality and when T_w is too large it may be difficult to find a sufficient number of windows with high-quality samples. Similarly, T_{max} should be larger than the maximum expected true offset. N_s should be sufficiently large to cover the search range with candidate sync points, but the only upper bound on its value is computational resources. In general, the optimal choice of these parameters will be dataset-dependent.

5 EXPERIMENTS AND RESULTS

We performed extensive experimentation in order to evaluate the proposed SyncWISE algorithm. We evaluated four different synchronization methods on two datasets: our novel S2S-Sync dataset and the CMU-MMAC kitchen activities dataset [16]. The algorithm variations are detailed in Sec. 5.1, followed by a discussion of error metrics in Sec. 5.2. Experiments on S2S-Sync and CMU-MMAC are detailed in Secs. 5.3 and 5.4, respectively.

5.1 Algorithm Variations

We identify four different synchronization methods that vary according to their treatment of the partial observability and coordinate registration problems. Our baseline method is the approach in [21], which performs a single global cross-correlation (CC). In the S2S-Sync dataset, we fill any missing accelerometry data by carrying over the last observation. We report results for two variants of this core approach.[6] The first is *Baseline-xx*, in which we select the x component of both the accelerometry and camera acceleration features, defined in Sec. 4.2.1. This choice was motivated by the orientation of the sensors (see Fig. 1(b)), as the x-axis corresponds to side-to-side motion and is most likely to result in a strong motion signal during movement in real-world settings.[7] The second variant is *Baseline-PCA*, for which we use the PCA-based approach described in Sec. 4.2.3 to determine the 1D signals used for CC. The SyncWISE family of methods uses the paired window matching approach with wKDE described in Sec. 4.3. The variant *SyncWISE-xx* uses the x-axis coordinate choice described above, while *SyncWISE* incorporates the PCA representation. *SyncWISE* is the best-performing variant across both datasets, and it establishes the new state-of-the-art for this problem. Note that in order to ensure a fair comparison, before applying cross-correlation to Baseline, we drop all low-quality windows from each clip and concatenate the high-quality windows together in both video and accelerometry, thereby ensuring that both methods see the same signals as input.

5.2 Evaluation Metric

We use two measures to evaluate algorithm performance: 1) the average of the absolute value of the synchronization error, E_{avg}, and 2) the percentage of clips which are synchronized to an offset error of less than n ms, PV-n. The choice of n connects to the question of how accurate temporal synchronization needs to be in order to be useful in practice. In general, the answer to this question will be application dependent. For example, a smoking puff can be as short as 500ms, but smoking sessions last 5-7 minutes [5] and teeth brushing lasts 2 min [1]. We note that the average annotator disagreement in S2S-Sync was 346ms, and prior works such as [8] used 300ms as the accuracy target. Based on these considerations, we chose to report *PV-300* and *PV-700* as the accuracy measures in our experiments. The PV-n measure is complementary to E_{ave}, which can be sensitive to outliers if a few difficult videos produce very large synchronization errors.

5.3 Experimental Results for S2S-Sync Dataset

We conducted two experiments to validate the performance of our method, using parameter settings described in Sec. 5.3.1. The experiment in Sec. 5.3.2 compares the performance of Baseline and SyncWISE using a simulated dataset with randomly-generated offsets, to provide a large-scale evaluation. The experiment in Sec. 5.3.3 demonstrates the ability of the method to synchronize the original clips in S2S-Sync using an iterative extension of our basic algorithm. In addition, Appendix A.3 describes a comprehensive sensitivity analysis on a held-out dataset which illustrates the impact of parameter changes on performance. All findings are discussed in Sec. 5.3.4.

[6] We used the open source reference implementation provided by the authors and incorporated it into our codebase so we could easily implement preprocessing and other steps. Our code and data are available at https://github.com/HAbitsLab/SyncWISE.
[7] We verified experimentally that selection of the y-axis direction results in worse performance.

Table 1. Result on S2S-Sync Dataset for baseline-xx, baseline-PCA, SyncWISE-xx and SyncWISE (T_w=10s, T_{max}=5s, N_s=20) with random shift. The SyncWISE results are averaged over 30 runs. In each run, we generate a random number from [-3 sec, 3 sec] as the ground truth shift between videos and accelerometer data. Baseline results are based on a single run because it is not affected by different input shift and no randomization is involved in this algorithm.

Method	# Videos	Ave #Win Pairs	Ave Error (ms)	PV-300 (%)	PV-700 (%)
Baseline-xx [21]	130	1	29690.79	50.77	82.31
Baseline-PCA	130	1	51124.77	47.69	70.77
SyncWISE-xx	130	1403.45	447.01	62.36	**89.72**
SyncWISE	130	1403.45	**416.30**	**73.38**	88.72

5.3.1 Parameter Specification. As described in Sec. 4.3.4, the choice of parameters for the method is dataset-dependent in general. We now detail the parameters used in all experiments in this section. The relatively high amount of missing data in the accelerometry signal motivated the choice of T_w = 10s. We performed a sensitivity analysis, detailed in Appendix A.3, to characterize the effect of N_s and T_{max} on the performance. We selected N_s = 20 and T_{max} = 5s. We used σ = 500 for the wKDE in Equation 4, and set the search bounds for μ and σ in obtaining confidences for Equation 6 to be $[-20,000ms, 20,000ms]$ and $[0, Inf]$, respectively.

5.3.2 SyncWISE Compared to Baseline. We applied a random 20/80 split of the 163 video clips, where 20% (33 videos) are used to optimize the parameters of our algorithm, and 80% (130 videos) are used to test our algorithm and produce the final results. Using the ground truth synchronization offsets for the 130 test video clips (see Sec. 3.3), we generate a synthetic testing dataset as follows: random offsets in the range [-3 sec, 3 sec] were sampled 30 times for each clip and used to shift the synchronization, resulting in 3,900 test clips. These synthesized test clips were used to assess the performance of our algorithm.[8] The experimental results are summarized in Table 1. We can see that E_{avg} is two orders of magnitude smaller for SyncWISE compared to Baseline. This is not surprising, as Baseline does not aggregate votes based on a quality measure and is therefore susceptible to the prevalent signal noise. Similarly, the PV-300 and PV-700 measures are higher for SyncWISE, although the difference is not as large, suggesting a smaller set of difficult clips may explain the high E_{avg} result for Baseline. We see that the use of PCA in SyncWISE provides a modest benefit relative to SyncWISE-xx, particularly for PV-300, but Baseline-PCA performs worse than Baseline-xx. We hypothesize that this is because the PCA step primarily benefits the quality of the confidence estimate, which is not used in Baseline. We include some examples of response curves for both methods in Appendix A.4.

5.3.3 Synchronizing with the Original Offsets. The simulation experiment in Sec. 5.3.2 facilitated a large-scale evaluation over a six second range of offsets. However, the ground truth offsets for the S2S-Sync clips are substantially larger and have a complex distribution, with an average offset of 21s, max offset of 180s, and min offset of 387ms. The direct application of SyncWISE to these clips is unsuccessful, as the search range is too large for accurate registration given the noisy and complex properties of these signals.

We therefore developed an interactive version of the SyncWISE method which can extend the search range arbitrarily, with the downside of requiring human intervention for verification. We make an analogy to image search to explain this issue. When the query is easy, the first returned result will have high confidence and can be accepted immediately. This is the case for synchronization with a six second offset range. But when the query is challenging, the confidence must be used to rank the results, and manual inspection is needed to verify the correct

[8]Since SyncWISE searches in the space of offsets and evaluates a discrete set of windows, it will produce different outputs for different relative shifts. In contrast, the Baseline approach uses a single global cross-correlation, and the shift-invariance property of the cross-correlation function means that the results will be the same for all shifts. Therefore, the Baseline method was run once for each clip.

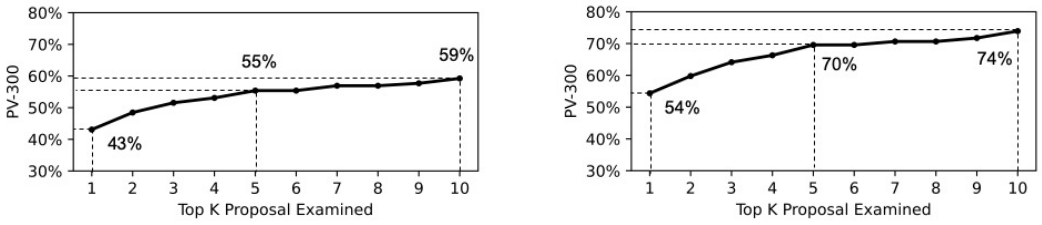


Fig. 5. Result of the Extended SyncWISE method on the S2S-Sync dataset with the original offsets (unsynchronized clips). Ranked proposals for sync points are examined to retrieve the correct sync. The plot shows the relationship between the number of top-ranked proposals examined and the resulting PV-300 measure (a) on 130 videos in test dataset, and (b) on 92 videos after removing problematic videos.

match. This is because the confidence estimates are not sufficiently accurate over an extremely large search range. We now describe the *Extended SyncWISE* approach as follows: We define a step size V_{step} of 3s and a maximum shift V_{max} of 3 min. These are chosen to search over the range of +/- 3 min, which is sufficient to cover all of the offsets in the S2S-Sync clips. We generate a set of shifted clips by starting with $-V_{max}$ and incrementing to $+V_{max}$ in steps of V_{step}, resulting in our case in 120 shifted clips. Each shifted clip is synchronized using SyncWISE, and the offsets from these clips are ranked according to the confidences computed via Equation 6. We then ask the user to examine the proposed sync points in ranked order, to identify a desired level of accuracy.

Since we already have the ground truth sync points for the purpose of this experiment, the role of the user can be automated. The results are depicted in Fig. 5. The x-axis shows the number K of top-K ranked proposal (candidate) sync points that must be examined, and the y-axis gives the percentage of clips correctly synchronized according to the PV-300 measure if the best proposal within the top K is selected. In the 130 videos in the test dataset, several had less than ideal recording conditions: 5 videos were recorded under dim lighting conditions, 1 video was blurred by water droplets on the lens; 9 videos had the edge of the lens covered by the GoPro case; and 23 exhibited little to no salient body movement. In Fig. 5 we report on both the 130 videos and the 92 videos (after removing these problematic videos). After removal of problematic videos, we can see that simply choosing the top-ranked sync proposal (as is done in the standard SyncWISE approach), results in a PV-300 of around 54%. However, examining the top 5 proposals permits an improvement to around 70%, while examining the top 10 proposals achieves a PV-300 of 74%. While this result falls short of a fully-automated approach, our method has the benefit of broad applicability and may still save significant human effort in synchronizing video clips under difficult and real-world recording conditions.

5.3.4 Discussion. Our results demonstrate the benefits of our SyncWISE approach over the Baseline global cross-correlation method. We believe this makes sense, as the baseline approach assumes that all parts of the video and accelerometry signals contain information which is relevant to the offset, and combines them into a single global estimate. This is unlikely to be true in our case due to partial observability. Our method attempts to automatically find most salient windows with high correlation to act as "marker gestures," similar to those defined manually in synchronizing data from a controlled in-lab study in [8, 34]. The problem of sychronizing clips over very long offsets in the presence of sensor noise remains open, but the Extended SyncWISE approach can provide a stop-gap interactive method. Use of SyncWISE requires the user to set the system parameters, and we included a sensitivity analysis in our experiments which leveraged the known ground truth. In general,

Table 2. Final result on the CMU-MMAC Dataset for synchronization between wearable camera and right arm accelerometor using SyncWISE, Baseline-PCA and Baseline-xx (with T_w=60s, T_{max}=60s, N_s=10).

Method	# Videos	Ave #Win Pairs	Ave Error (ms)	PV300	PV700
Baseline-xx [21]	126	1	26371.69	30.16	43.65
Baseline-PCA	126	1	21983.33	40.48	50.0
SyncWISE	126	2865.32	**14098.41**	**49.21**	**64.29**

it will be difficult to specify the correct system parameters in advance for all sync problems. In practice, we believe an iterative approach will be needed, in which an initial set of parameters are refined through repeated synchronization and analysis of a small set of clips. Once effective parameters are found, then an entire dataset can be synchronized automatically, or interactively using Extended SyncWISE.

5.4 Experimental Results for CMU-MMAC Dataset

We now describe our experiments on CMU-MMAC, a well-known multimodal activity dataset [16]. The data of different modalities used in this experiment are described in Sec. 5.4.1. Parameter settings are described in Sec. 5.4.2. The experiment in Sec. 5.4.3 compares the performance of Baseline and SyncWISE using the ground truth synchronization provided with the dataset. In addition, Appendix A.5 provides additional results and examples. The findings are discussed in Sec. 5.4.4.

5.4.1 CMU-MMAC Dataset. In this dataset, a wearable camera is attached to a head lamp whose bulb has been removed and it is worn around the head of the subject, and five accelerometer sensors are placed on the subject's back, legs, and arms. Data is recorded by a laptop through wired connections, and synchronization between the signals is achieved through network time sync protocols. The ground truth offsets to achieve synchronization are within the range of [1.8s, 178.0s], with a mean of 26.4s. The videos are collected at 30 fps and accelerometry data at 125 Hz. There are 126 sessions from 30 subjects which have valid video and IMU data. In this experiment, we use the unsychronized video and accelerometry as the input to our method.

5.4.2 Parameter Specification. We interpolate the accelerometry data to match the video sampling rate. We choose random offsets o_j^i over three ranges, from 0 to 60, 90, or 120, to generate an initial offset for synchronizing and then apply windowed cross-correlation. Note that the maximum temporal offset we can recover with our method is the sum of the maximum random offset amount T_{max} and the window size T_w. In this dataset, we set $N_s = 10$. We select a Gaussian kernel $K(\delta, t)$ with $\sigma = 3000ms$. We set the search bounds for estimating μ and σ of $g(t)$ to be $[-600s, 600s]$ and $[0, Inf]$, respectively.

5.4.3 SyncWISE Compared to Baseline. Table 2 shows the results from the Baseline method and the SyncWISE algorithm. The input clips combine the wearable camera signal with the accelerometer data coming from the right arm. We include the results for pairing video with the accelerometers from the other locations, along with different parameter combinations and examples of the response curves from both methods, in Appendix A.5. This dataset is challenging because none of the accelerometery sensors are co-located with the camera. The frequent relative movement between the sensor and camera creates challenges for the baseline method. From Table S2, we can see that PCA improves the performance of the baseline, presumably by identifying the corresponding axes across modalities in the face of relative movement of the sensors. Our window-weighted kernel density estimation approach further improves the performance consistently across all of the different accelerometer positions. This demonstrates the ability of our method to automatically focus on the "signal intensive" temporal windows and ignore those of little synchronization utility.

5.4.4 Discussion. In the case of the CMU-MMAC dataset, where data is collected with a fixed sampling rate and no data frames are dropped during capture, we are able to select a large window size T_w without worrying about problems arising due to low data quality, and use a reasonable empirically set value for T_{max}. At the same time, the performance of all methods on this dataset are lower than that of the S2S-Sync dataset, which suggests that the large relative motion between the camera and the accelerometers, as a result of their positions on the body, may be playing a role in making the sync task quite challenging. The experiments in Appendix A.5 further demonstrate that parameter choices are dataset-dependent. Even for the same dataset, the best performance for accelerometers located at different positions is achieved by different parameter choices.

6 DISCUSSION, CONCLUSION AND FUTURE WORK

The ability to accurately synchronize multiple data streams is a longstanding problem with increasing utility for researchers who seek to develop and validate computational models to detect daily human behaviors from multimodal wearable sensor suites in the wild. Specifically, as visual confirmation is a widely used method to label target events, an accurate, feasible, less burdensome means to synchronize video with adjacent sensor streams has become urgently needed.

In the current study, we demonstrated the feasibility and effectiveness of a novel approach (SyncWISE) to synchronize data from a wearable video camera with data from a wearable accelerometer. SyncWISE addresses the problems of partial observability (where sensors do not capture the same events) and coordinate transformation (sensor axes are not spatially-aligned over time) that characterize challenging real-world synchronization tasks involving data collected from wearable sensors. We evaluate SyncWISE on two datasets: a novel smoking cessation dataset S2S-Sync and the CMU-MMAC dataset [16]. We demonstrate state-of-the-art performance relative to a recent baseline method.

Although our work focuses on resolving the time synchronization problem between a wearable camera and a chest-worn accelerometer sensor, the algorithm design can be further adapted to other sensing modalities on different parts of the human body. Doing so will enable temporal alignment of video-derived labels to diverse wearable sensor signals. Such temporally-precise labels obtained from the in-wild environment can improve the accuracy of detecting fine-grained micro-behaviors (e.g., dynamics of hand-to-mouth gestures) underlying a wide variety of daily behaviors such as eating, drinking, brushing, flossing, and smoking. In addition to advancing computational models for detecting daily behaviors, fine-grained observations of micro-behaviors in the wild can also advance our understanding of daily human behaviors. Finally, we provide our novel S2S-Sync dataset to the research community, for use as a benchmark for in-the-wild time synchronization, along with our time synchronization software, including scripts for reproducing all of the experiments in this manuscript, so as to continue to advance this area of research.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable suggestions for improving the manuscript. We also thank Shahin Samiei for IRB and data management support, and Dr. Timothy Hnat and Dr. Monowar Hossain for software support. Research reported here was supported by the National Institutes of Health (NIH) under award K25DK113242 (by NIDDK) and U54EB020404 (by NIBIB) through funds provided by the trans-NIH Big Data-to-Knowledge (BD2K) initiative. We would also like to acknowledge support by the National Science Foundation (NSF) under awards CNS1915847 and CNS1823201. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NIH or the NSF.

REFERENCES

[1] Sayma Akther, Nazir Saleheen, Shahin Alan Samiei, Vivek Shetty, Emre Ertin, and Santosh Kumar. 2019. MORAL: An MHealth Model for Inferring Oral Hygiene Behaviors in-the-Wild Using Wrist-Worn Inertial Sensors. *Proc. ACM Interact. Mob. Wearable Ubiquitous*

Technol. 3, 1, Article 1 (March 2019), 25 pages. https://doi.org/10.1145/3314388

[2] Rawan Alharbi, Angela Pfammatter, Bonnie Spring, and Nabil Alshurafa. 2017. WillSense: Adherence Barriers for Passive Sensing Systems That Track Eating Behavior. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI EA '17)*. Association for Computing Machinery, New York, NY, USA, 2329–2336. https://doi.org/10.1145/3027063.3053271

[3] Rawan Alharbi, Tammy Stump, Nilofar Vafaie, Angela Pfammatter, Bonnie Spring, and Nabil Alshurafa. 2018. I Can't Be Myself: Effects of Wearable Cameras on the Capture of Authentic Behavior in the Wild. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3, Article 90 (Sept. 2018), 40 pages. https://doi.org/10.1145/3264900

[4] Nabil Alshurafa, Annie Wen Lin, Fengqing Zhu, Roozbeh Ghaffari, Josiah Hester, Edward Delp, John Rogers, and Bonnie Spring. 2019. Counting Bites With Bits: Expert Workshop Addressing Calorie and Macronutrient Intake Monitoring. *J Med Internet Res* 21, 12 (4 Dec 2019), e14904. https://doi.org/10.2196/14904

[5] Ahsan Amin, Ali, Syed Monowar Hossain, Karen Hossain, Md Hovsepian, Md Rahman, Kurt Plarre, and Santosh Kumar. 2012. mPuff: Automated Detection of Cigarette Smoking Puffs from Respiration Measurements. In *2012 ACM/IEEE 11th International Conference on Information Processing in Sensor Networks (IPSN)*. 269–280.

[6] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. 2016. Soundnet: Learning Sound Representations from Unlabeled Video. In *Advances in neural information processing systems*. 892–900.

[7] Jum-Han Bae and Jong-Tae Kim. 2015. Design and Implementation of High Accurate Synchronization between Gyroscope and Image Sensor. In *2015 Seventh International Conference on Ubiquitous and Future Networks*. IEEE, 956–958.

[8] David Bannach, Oliver Amft, and Paul Lukowicz. 2009. Automatic Event-based Synchronization of Multimodal Data Streams from Wearable and Ambient Sensors. *European Conference on Smart Sensing and Context* 5741 (2009), 135–148.

[9] Shengjie Bi, Tao Wang, Nicole Tobias, Josephine Nordrum, Shang Wang, George Halvorsen, Sougata Sen, Ronald Peterson, Kofi Odame, Kelly Caine, and et al. 2018. Auracle: Detecting Eating Episodes with an Ear-Mounted Sensor. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3, Article 92 (Sept. 2018), 27 pages. https://doi.org/10.1145/3264902

[10] Brandon M Booth, Karel Mundnich, Tiantian Feng, Amrutha Nadarajan, Tiago H Falk, Jennifer L Villatte, Emilio Ferrara, and Shrikanth Narayanan. 2019. Multimodal Human and Environmental Sensing for Longitudinal Behavioral Studies in Naturalistic Settings: Framework for Sensor Selection, Deployment, and Management. *J Med Internet Res* 21, 8 (20 Aug 2019), e12832. https://doi.org/10.2196/12832

[11] Hennie Brugman and Albert Russel. 2004. Annotating Multi-media/Multi-modal Resources with ELAN. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*. European Language Resources Association (ELRA), Lisbon, Portugal. http://www.lrec-conf.org/proceedings/lrec2004/pdf/480.pdf

[12] Shanshan Chen, Jeff S. Brantley, Taeyoung Kim, and John Lach. 2010. Characterizing and Minimizing Synchronization and Calibration Errors in Inertial Body Sensor Networks. In *Proceedings of the Fifth International Conference on Body Area Networks* (Corfu, Greece) *(BodyNets '10)*. 138–144. https://doi.org/10.1145/2221924.2221951

[13] Joon Son Chung and Andrew Zisserman. 2016. Out of Time: Automated Lip Sync in the Wild. In *Asian conference on computer vision*. Springer, 251–263.

[14] Enea Cippitelli, Samuele Gasparrini, Ennio Gambi, Susanna Spinsante, Jonas Wåhslény, Ibrahim Orhany, and Thomas Lindhy. 2015. Time Synchronization and Data Fusion for RGB-depth Cameras and Inertial Sensors in AAL Applications. In *2015 IEEE International Conference on Communication Workshop (ICCW)*. IEEE, 265–270.

[15] The SciPy community. 2019. SciPy Library Curve Fitting Method. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html. Accessed: 2020-05-15.

[16] Fernando de la Torre, Jessica K. Hodgins, Javier Montano, and Sergio Valcarcel. 2009. Detailed Human Data Acquisition of Kitchen Activities: the CMU-Multimodal Activity Database (CMU-MMAC). In *CHI 2009 Workshop. Developing Shared Home Behavior Datasets to Advance HCI and Ubiquitous Computing Research.*

[17] Emre Ertin, Nathan Stohs, Santosh Kumar, Andrew Raij, Mustafa al'Absi, and Siddharth Shah. 2011. AutoSense: Unobtrusively Wearable Sensor Suite for Inferring the Onset, Causality, and Consequences of Stress in the Field. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems* (Seattle, Washington) *(SenSys '11)*. Association for Computing Machinery, New York, NY, USA, 274–287. https://doi.org/10.1145/2070942.2070970

[18] Adam Fouse, Nadir Weibel, Edwin Hutchins, and James Hollan. 2011. Chronoviz: A System for Supporting Navigation of Time-coded Data. *Conference on Human Factors in Computing Systems - Proceedings*, 299–304. https://doi.org/10.1145/1979742.1979706

[19] Franklin, Aretha Retrieved February, 2020. Amazing Grace Wikipedia Website. https://en.wikipedia.org/wiki/Amazing_Grace_(2018_film).

[20] Ido Freeman, Patrick Wieschollek, and Hendrik P. A. Lensch. 2016. Robust Video Synchronization using Unsupervised Deep Learning. *CoRR* abs/1610.05985 (2016). arXiv:1610.05985 http://arxiv.org/abs/1610.05985

[21] Lex Fridman, Daniel E Brown, William Angell, Irman Abdić, Bryan Reimer, and Hae Young Noh. 2016. Automated Synchronization of Driving Data using Vibration and Steering Events. *Pattern Recognition Letters* 75 (2016), 9–15.

[22] Yi Chiew Han, Kiing Ing Wong, and Iain Murray. 2019. Automatic Synchronization of Markerless Video and Wearable Sensors for Walking Assessment. *IEEE Sensors Journal* (2019).
[23] David Harwath, Antonio Torralba, and James Glass. 2016. Unsupervised Learning of Spoken Language with Visual Context. In *Advances in Neural Information Processing Systems*. 1858–1866.
[24] Timothy Hnat, Syed Hossain, Nasir Ali, Simona Carini, Tyson Condie, Ida Sim, Mani Srivastava, and Santosh Kumar. 2017. mCerebrum and Cerebral Cortex: A Real-time Collection, Analytic, and Intervention Platform for High-frequency Mobile Sensor Data. In *AMIA (American Medical Informatics Association) 2017 Annual Symposium*. American Medical Informatics Association.
[25] Syed Monowar Hossain, Timothy Hnat, Nazir Saleheen, Nusrat Jahan Nasrin, Joseph Noor, Bo-Jhang Ho, Tyson Condie, Mani Srivastava, and Santosh Kumar. 2017. MCerebrum: A Mobile Sensing Software Platform for Development and Validation of Digital Biomarkers and Interventions. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems* (Delft, Netherlands) *(SenSys '17)*. Association for Computing Machinery, New York, NY, USA, Article 7, 14 pages. https://doi.org/10.1145/3131672.3131694
[26] Karen Hovsepian, Mustafa al'Absi, Emre Ertin, Thomas Kamarck, Motohiro Nakajima, and Santosh Kumar. 2015. CStress: Towards a Gold Standard for Continuous Stress Assessment in the Mobile Environment. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Osaka, Japan) *(UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 493–504. https://doi.org/10.1145/2750858.2807526
[27] Jacob W. Kamminga, Lara M. Janßen, Nirvana Meratnia, and Paul J. M. Havinga. 2019. Horsing Around—A Dataset Comprising Horse Movement. *Data* 4, 4 (2019). https://doi.org/10.3390/data4040131
[28] Sami M. Lasassmeh and James M. Conrad. 2010. Time Synchronization in Wireless Sensor Networks: A survey. In *Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon)*. 242–245. https://doi.org/10.1109/SECON.2010.5453878
[29] Yin Li, Miao Liu, and James M Rehg. 2018. In the eye of beholder: Joint learning of gaze and actions in first person video. In *ECCV*.
[30] Timecode Systems Limited. 2020. Timecode Systems. https://www.timecodesystems.com/syncbac-pro/. Accessed: 2020-05-15.
[31] Miao Liu, Xin Chen, Yun Zhang, Yin Li, and James M Rehg. 2020. Attention Distillation for Learning Video Representations. In *BMVC*.
[32] MD2K. 2020. MotionSense - MD2K. https://md2k.org/documentation/data_dictionary/raw_streams/motionsense.html. Accessed: 2020-05-15.
[33] Peter Nilsson, John-Olof; Händel. 2010. Time Synchronization and Temporal Ordering of Asynchronous Sensor Measurements of a Multi-sensor Navigation System. In *IEEE/ION Position, Location and Navigation Symposium*. 897–902.
[34] Thomas Ploetz, Chen Chen, Nils Hammerla, and Gregory Abowd. 2012. Automatic Synchronization of Wearable Sensors and Video-Cameras for Ground Truth Annotation — A Practical Approach. *Proceedings - International Symposium on Wearable Computers, ISWC*. https://doi.org/10.1109/ISWC.2012.15
[35] Valentin Radu and Maximilian Henne. 2019. Vision2Sensor: Knowledge Transfer Across Sensing Modalities for Human Activity Recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3, Article 84 (Sept. 2019), 21 pages. https://doi.org/10.1145/3351242
[36] Nazir Saleheen, Amin Ahsan Ali, Syed Monowar Hossain, Hillol Sarker, Soujanya Chatterjee, Benjamin Marlin, Emre Ertin, Mustafa al'Absi, and Santosh Kumar. 2015. PuffMarker: A Multi-Sensor Approach for Pinpointing the Timing of First Lapse in Smoking Cessation. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Osaka, Japan) *(UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 999–1010. https://doi.org/10.1145/2750858.2806897
[37] Peter Sand and Seth Teller. 2004. Video Matching. In *ACM SIGGRAPH 2004 Papers* (Los Angeles, California) *(SIGGRAPH '04)*. Association for Computing Machinery, New York, NY, USA, 592–599. https://doi.org/10.1145/1186562.1015765
[38] F. Sivrikaya and B. Yener. 2004. Time Synchronization in Sensor Networks: A Survey. *IEEE Network* 18, 4 (July 2004), 45–50. https://doi.org/10.1109/MNET.2004.1316761
[39] Isaac Skog and Peter Handel. 2011. Time Synchronization Errors in Loosely Coupled GPS-Aided Inertial Navigation Systems. *IEEE Transactions on Intelligent Transportation Systems* 12, 4 (Dec 2011), 1014–1023. https://doi.org/10.1109/TITS.2011.2126569
[40] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. 2018. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 8934–8943.
[41] Oliver Wang, Christopher Schroers, Henning Zimmer, Markus Gross, and Alexander Sorkine-Hornung. 2014. VideoSnapping: Interactive Synchronization of Multiple Videos. *ACM Trans. Graph.* 33, 4, Article 77 (July 2014), 10 pages. https://doi.org/10.1145/2601097.2601208
[42] Ethan Z. Welty, Timothy C. Bartholomaus, Shad O'Neel, and W. Tad Pfeffer. 2013. Cameras as Clocks. *Journal of Glaciology* 59, 214 (2013), 275–286. https://doi.org/10.3189/2013JoG12J126
[43] Patrick Wieschollek, Ido Freeman, and Hendrik PA Lensch. 2017. Learning Robust Video Synchronization without Annotations. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 92–100. https://doi.org/10.1109/ICMLA.2017.0-173
[44] Xitong Yang, Palghat Ramesh, Radha Chitta, Sriganesh Madhvanath, Edgar A Bernal, and Jiebo Luo. 2017. Deep Multimodal Representation Learning from Temporal Data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5447–5455.
[45] Shibo Zhang, Yuqi Zhao, Dzung Tri Nguyen, Runsheng Xu, Sougata Sen, Josiah Hester, and Nabil Alshurafa. 2020. NeckSense: A Multi-Sensor Necklace for Detecting Eating Activities in Free-Living Conditions. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 2, Article 72 (June 2020), 26 pages. https://doi.org/10.1145/3397313

[46] Yun C. Zhang and James M. Rehg. 2018. Watching the TV Watchers. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 2, Article 88 (July 2018), 27 pages. https://doi.org/10.1145/3214291

A APPENDIX

A.1 Data Filtering

Because participants can turn on and off the camera at any time, the video clips can be of any length. After removing the clips shorter than 30 seconds, we received a total number of 378 on-body GoPro video clips. We present our dataset screening procedure in a flow diagram in Fig. S1. The 378 video clips go through an initial video quality screening phase where 31 video clips are excluded due to poor lighting condition affecting visibility of the recorded video footage, and one extra video clip is removed because the lens was blocked by the wearer's clothing. We filtered data due to sensor quality and hardware error. Since our approach requires a minimum number (20) of 10-second windows with high-quality data (as discussed in Sec. 4.2), we exclude 136 clips that do not meet this minimum threshold of data quality. Additionally, when we visualized the sensor data together with the video, we discovered two video clips with erroneous sensor readings, which resulted from a faulty device. We then removed video clips that did not have a landmark or distinguishable human movement (needed to manually synchronize between the two signals, and explained in Sec. 3.3). After filtering videos that are not capable of being synchronized, we end up with 163 video clips (45.2 hours) and sensor data from a total of 21 participants. Each participant contributed 2.15 hours on average of "usable" data for analysis.

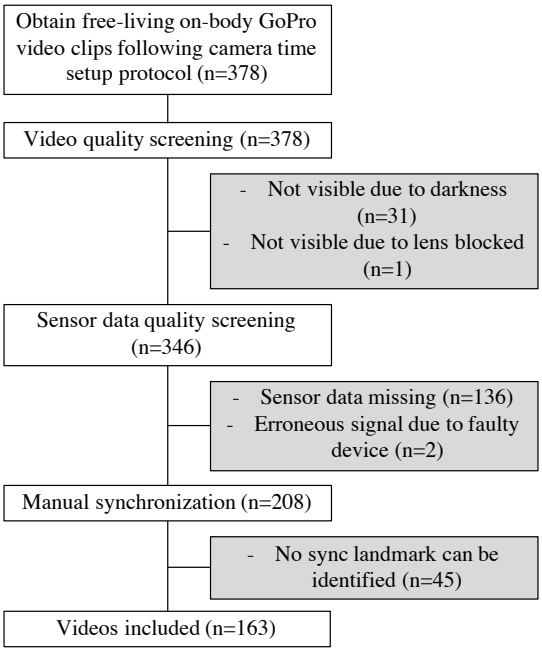


Fig. S1. Flow diagram of dataset formation.

A.2 SyncWISE Algorithm

Algorithm 1: SyncWISE: Synchronization based on Window Induced Shift Estimation

input : video and accelerometer data clip with asynchronous clocks
output: time shift δ of video and accelerometer data and confidence C
Obtain optical flow of video;
Calculate 1-component PCA of optical flow as $x_1, x_2, \ldots, x_k$ and 3-axis accelerometer data as $y_1, y_2, \ldots, y_k$;
Data screening on accelerometer data to obtain T_w second windows of high data quality, and resample to the same sampling rate as the optical flow signal;
Initialize N_s, T_{max}, T_w;
$N \leftarrow$ number of windows;
for *i-th window* $(s_i, s_i + T_w)$ **do**
  for $j = 1, 2, \ldots, N_s$ **do**
    $o_j \leftarrow random(-T_{max}, T_{max})$;
    $\vec{x} \leftarrow x_{s_i+o_j}, x_{s_i+1+o_j}, \ldots, x_{s_i+T_w+o_j}$;
    $\vec{y} \leftarrow y_{s_i}, y_{s_i+1}, \ldots, y_{s_i+T_w}$;
    $cc(\tau) \leftarrow$ cross-correlation of $\vec{x}$ and $\vec{y}$;
    $\delta_{ij} \leftarrow \arg\max_\tau \left| cc_{i,j}(\tau) - \underset{t\in[-T_w,T_w]}{\text{median}} \{cc_{i,j}(t)\} \right| + o_j^i$;
    $conf_{ij} \leftarrow \frac{\max_\tau \left| cc_{i,j}(\tau) - \underset{t\in[-T_w,T_w]}{\text{median}} \{cc_{i,j}(t)\} \right|}{\underset{t\in[-T_w,T_w]}{\text{std}} \{cc_{i,j}(\tau)\}}$;
  end
end
$f(t) \leftarrow 0$;
for $i = 1, 2, \ldots, N$ **do**
  for $j = 1, 2, \ldots, N_s$ **do**
    $f(t) \leftarrow f(t) + conf_{ij} \cdot K(\delta_{ij}, t), K() = Gaussian()$;
  end
end
$f(t) \leftarrow \frac{1}{\sum_i \sum_j conf_{ij}} f(t)$;
$\delta \leftarrow \arg\max_t f(t)$;
Fit Gaussian curve $\mathcal{N}(\mu, \sigma)$ to $f(t)$;
$C \leftarrow \frac{Const}{\hat{\sigma} \cdot var(\hat{\mu})}$

A.3 Parameter Optimization on S2S-Sync Dataset

We perform sensitivity analysis (parameter optimization) using 20% (33 of the 163) of the videos. We set the Gaussian kernel σ to 500. We also optimize the following 3 main parameters (T_w, T_{max}, and N_s).

A.3.1 Optimization for Window Size T_w. As described in Sec. 4.2.1, since we eliminate the seconds with low data quality, if T_w is too large, then it may disqualify several windows due to the 80% sampling rate quality threshold. When we set T_w as 10 s, after the data screening step, the 33 videos each have on average 82 qualified high-quality windows. When we adjust T_w to 20s, 29 videos no longer have a sufficient number of qualified windows. After

Table S1. Parameter T_{max} search (T_w=10s, N_s=20) for S2S-Sync dataset

Input Shift (s)	T_{max} (s)	Ave Error (ms)	PV-700	PV-300	Ave Conf.
0	2	268	0.94	0.79	892
0	4	**262**	0.94	0.79	96
0	6	284	0.94	0.76	11
0	8	364	0.94	0.76	1
0	10	282	0.94	0.76	0
2	2	447	0.91	0.79	106
2	4	307	0.91	0.79	18
2	6	**264**	0.91	0.76	5
2	8	597	0.88	0.73	1
2	10	753	0.88	0.67	0
4	2	947	0.79	0.67	1
4	4	923	0.82	0.7	1
4	6	786	0.85	0.73	0
4	8	814	0.88	0.73	0
4	10	**716**	0.88	0.76	0

more investigation we set the window size to 10s, to ensure enough videos can be processed, and we can address low-quality video-sensor pairs.

A.3.2 Optimization for Max Random Offset T_{max}. We tested a range of values for T_{max} from 2s to 10s with input data of different shifts from 0 to 4s. As shown in Table S1, when the input shift is 2 and T_{max} is set to 6s, the average error is 264ms. The average error goes up to 753ms with increasing T_{max}. When the input shift is 0s, the average errors are 268, 262, and 284ms when T_{max} is 2s, 4s, and 6s, respectively, showing no significant difference. With a 4s input shift, the minimum average error occurs when T_{max} is set to 10s. When we apply SyncWISE in a real world setting, due to a lack of the prior knowledge of the input shift, we select T_{max} to be 5s (average between 4 and 6), as long as the target offset range is less than 4s.

A.3.3 Optimization for Number of Random Offsets N_s. We then fixed T_w and T_{max} and adjusted N_s. As shown in Fig. S2, when we adjusted N_s in the range from 20 to 80, with input data of different shifts of 0 and 2s, the average error did not significantly change, suggesting that 20 offsets within a window is sufficient. Fig. S2 shows the average error as a function of N_s. We selected N_s to be 20, which provides the least average error.

A.4 Examples in S2S-Sync Dataset

A.4.1 Examples of Response Curves. Figure S3 shows example response curves of both methods on two sessions using the wearable camera and chest accelerometer. Baseline method fails in the second session because of the irregular sampling rate and sparse signal segments in our dataset.

A.4.2 Examples of Confidence Scores. Figure S4 shows some examples of voted shift curves $f(t)$ and their corresponding confidence score. Confidence in (a) and (b) is low due to large variance and spurious peaks. Figure(c) shows an example of high confidence.

A.5 Results on CMU-MMAC Dataset

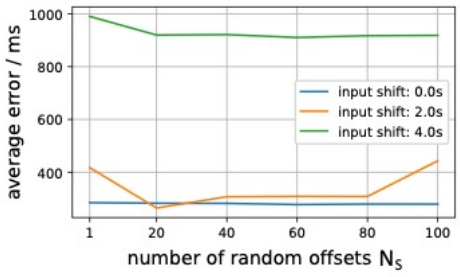
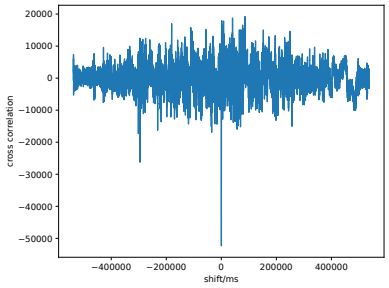
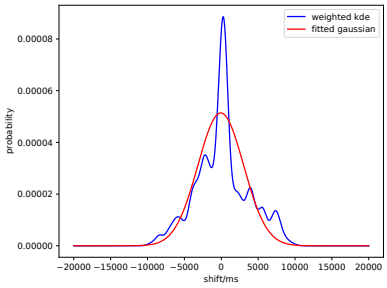


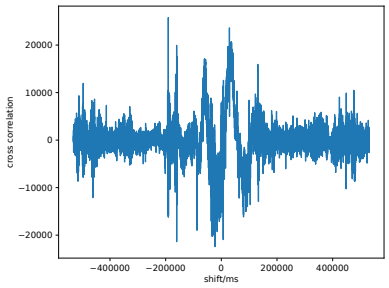
Fig. S2. Average error with different number of random offsets for each window (T_w=10s, T_{max}=3s).



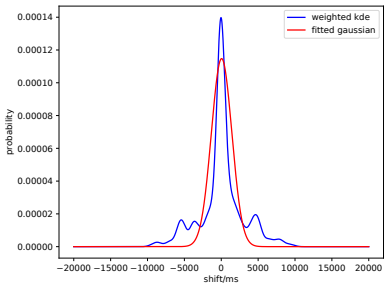
(a) Baseline-xx estimation on session 1 is -267ms



(b) SyncWISE estimation on session 1 is 261ms



(c) Baseline-xx estimation on session 2 is 190426ms

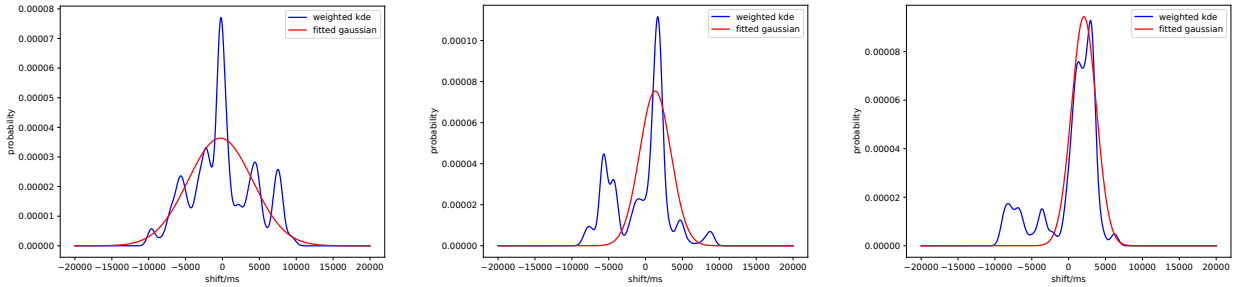


(d) SyncWISE estimation on session 2 is -34ms

Fig. S3. Example response curves of both methods on two sessions using a wearable camera and chest accelerometer. Ground truth for both sessions is 0s. (a) and (c) show the cross-correlation function of baseline method. (b) and (d) show $f(t)$ of SyncWISE for the same sessions. In (b), 480 windows are sampled and the confidence score of the estimation is 0.89. In (d), 2140 windows are sampled and the confidence score of the estimation is 10.11.

A.5.1 Results on all IMU Positions and Different Parameter Combinations. As shown in Table S3, among all window size/maximum random offset configurations, SyncWise-60/60 achieves the best performance. This supports our analysis in Sec. 4.3.4 that large window sizes and small maximum random offsets are preferred in this algorithm.

Comparing results across different IMU positions, we find that right arm acceleration is most informative in synchronizing with the camera using both approaches followed by the back sensor, with the left and right leg data



(a) Ground truth shift = -325ms (b) Ground truth shift = 1827ms (c) Ground truth shift = 2755ms

Fig. S4. Examples of video shift and confidence estimation. In (a), 500 windows are sampled, SyncWISE estimation is -206 ms and the confidence score of the estimation is 0.19. In (b), 260 windows are sampled, SyncWISE estimation is 1651 ms and the confidence score of the estimation is 1.13. In (c), 460 windows are sampled, SyncWISE estimation is 2958 ms and the confidence score of the estimation is 9.37. Confidence in (a) and (b) is low due to large variance and spurious votings. Figure(c) shows an example of high confidence. The fitted Gaussian successfully finds the true peak and ignores the false peaks.

Table S2. Baseline results on CMU-MMAC

IMU Position	Baseline-xx			Baseline-PCA		
	Average Error	PC-300	PC-700	Average Error	PC-300	PC-700
Left Arm	32687.30	12.70	16.67	29832.01	20.63	30.16
Right Arm	26371.69	30.16	43.65	21983.33	40.48	50.00
Left Leg	50740.48	2.38	7.14	46670.90	7.94	13.49
Right Leg	52933.07	1.59	4.76	49400.26	3.97	8.73
Back	53221.43	3.17	5.56	29112.96	23.81	30.95

being the least informative modalities. This is consistent with our findings. In this kitchen dataset, the subjects perform a lot of actions using their right hand which cause the camera to vibrate accordingly. In contrast, leg motion has less impact on the camera located on the head of subjects. This is an interesting finding because we expected that the IMU on the subject's back had the least relative motion to the camera and as a result would yield the best synchronization performance. But the results show that variability in the data, representing frequent motion, is essential for successful synchronization, even when the camera and accelerometer are not co-located.

A.5.2 Comparing Window Sampling Methods. We randomly selected 10 videos from the CMU-MMAC dataset to validate our choice of random offset instead of evenly placed offsets. Table S4 shows results using even spaced sampling and random sampling. The window size and maximum search range are set at 60s and 60s, respectively. $N_s = 20$ window pairs are sampled for each sliding window in accelerometer data. The two sampling methods do not yield significant performance difference.

A.5.3 Examples of Response Curves in CMU-MMAC Dataset. Figure S5 shows example response curves of both methods on two sessions using the wearable camera and right arm accelerometer in the CMU-MMAC dataset.

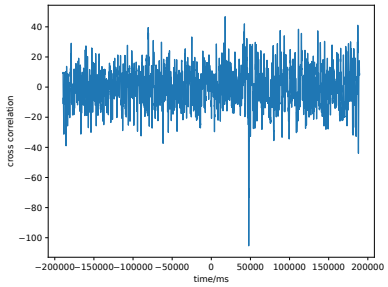
Table S3. SyncWISE results on CMU-MMAC

IMU Position	SyncWise-30/60			SyncWise-30/90			SyncWise-30/120		
	Ave Error	PV-300	PV-700	Ave Error	PV-300	PV-700	Ave Error	PV-300	PV-700
Left Arm	19539.15	22.22	43.65	20157.14	24.60	43.65	26284.92	19.84	33.33
Right Arm	15902.12	43.65	**65.87**	17626.72	39.68	60.32	19237.83	34.13	53.17
Left Leg	19738.36	22.22	42.06	20122.22	12.70	39.68	23418.52	11.90	29.37
Right Leg	16624.07	20.63	42.06	20339.68	12.70	36.50	21445.50	13.49	30.95
Back	16755.03	43.65	59.52	18342.59	38.88	56.35	18522.22	38.89	53.97
IMU Position	SyncWise-60/60			SyncWise-60/90			SyncWise-60/120		
	Ave Error	PV-300	PV-700	Ave Error	PV-300	PV-700	Ave Error	PV-300	PV-700
Left Arm	16808.99	25.40	43.65	17866.67	24.60	44.44	18337.30	22.22	40.48
Right Arm	14358.99	**49.21**	63.49	13194.71	46.83	62.70	12669.58	42.06	61.11
Left Leg	20927.25	10.32	28.57	19902.38	11.90	31.75	25044.44	9.52	26.98
Right Leg	20777.78	13.49	32.54	22119.58	11.11	27.78	22855.29	9.52	25.40
Back	15759.26	39.68	53.17	16034.39	35.71	52.38	15425.13	34.13	53.17

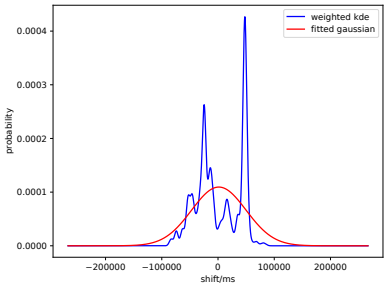
Table S4. Experiment on window sampling methods

	Even Spaced Sampling			Random Sampling		
IMU position	Ave error	PV-300	PV-700	Ave error	PV-300	PV-700
Left Arm	25526.67	0	20	29263.33	0	30
Right Arm	38446.67	50	50	38433.33	50	50
Left Leg	35470.00	20	20	35486.67	20	30
Right Leg	46040.00	10	20	39883.33	10	20
Back	29760.00	40	50	29776.67	40	50

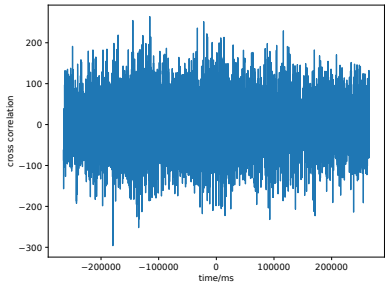
Both methods work well in the first session. Baseline method failed in the second session because this session contains less hand movement with high frequency and high magnitude across the video.



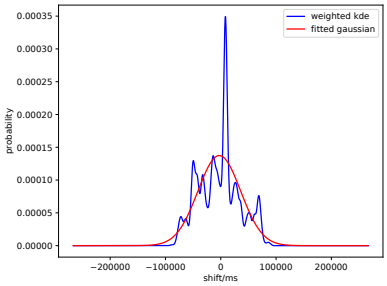
(a) Baseline-PCA estimation on session 1 is 47.800s



(b) SyncWISE-60/60 estimation on session 1 is 47.733s



(c) Baseline-PCA estimation on session 2 is -179.433s



(d) SyncWISE-60/60 estimation on session 2 is 8.400s

Fig. S5. Example response curves of both methods on two sessions using wearable camera and right arm accelerometer. (a) & (b) Ground truth 47.700s, (c) & (d) Ground truth 8.000s. In (b), 3190 windows are sampled and the confidence score of the estimation is 0.05. In (d), 4710 windows are sampled and the confidence score of the estimation is 0.39.