De-anonymizing VR Avatars using Non-VR Motion Side-channels

Mohd Sabra
University of Texas at San Antonio
San Antonio, TX, USA
mohd.sabra@utsa.edu

Nisha Vinayaga-Sureshkanth
University of Texas at San Antonio
San Antonio, TX, USA
vsnisha@ieee.org

Ari Sharma
University of Texas at Austin
Austin, TX, USA
arisharma@utexas.edu

Anindya Maiti
University of Oklahoma
Norman, OK, USA
am@ou.edu

Murtuza Jadliwala
University of Texas at San Antonio
San Antonio, TX, USA
murtuza.jadliwala@utsa.edu

ABSTRACT

Virtual Reality (VR) technology offers an immersive audio-visual experience to users through which they can interact with a digitally represented 3D space (i.e., a *virtual world*) using a headset device. By (visually) transporting users from their physical world to realistic virtual spaces, VR systems enable interactive and true-to-life versions of traditional applications such as gaming, remote conferencing and virtual tourism. However, VR applications also present significant user-privacy challenges. This paper studies a new type of privacy threat targeting VR users which attempts to connect their activities visible in the virtual world to their physical state sensed in the real world. Specifically, this paper analyzes the feasibility of carrying out a *de-anonymization* or *identification* attack on VR users by *correlating* visually observed movements of users' avatars in the virtual world with some auxiliary data (e.g., motion sensor data from mobile/wearable devices) representing their context/state in the physical world. To enable this attack, the paper proposes a novel framework which first employs a learning-based activity classification approach to translate the disparate visual movement data and motion sensor data into an *activity-vector* to ease comparison, followed by a filtering and identity ranking phase outputting an ordered list of potential identities corresponding to the target visual movement data. A comprehensive empirical evaluation of the proposed framework is conducted to study the feasibility of such a de-anonymization attack.

CCS CONCEPTS

• **Security and privacy**;

KEYWORDS

Virtual reality, motion, side channel, de-anonymization.

ACM Reference Format:
Mohd Sabra, Nisha Vinayaga-Sureshkanth, Ari Sharma, Anindya Maiti, and Murtuza Jadliwala. 2024. De-anonymizing VR Avatars using Non-VR Motion Side-channels . In *Proceedings of the 17th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '24), May 27–30, 2024, Seoul, Republic of Korea.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3643833.3656135

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

WiSec '24, May 27–30, 2024, Seoul, Republic of Korea
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0582-3/24/05
https://doi.org/10.1145/3643833.3656135

1 INTRODUCTION

Virtual Reality (VR) is a transformative technology which has fundamentally changed how we interact with digital spaces. By utilizing specialized hardware such as VR headsets and input controllers, users can immerse themselves in three-dimensional, computer-generated realistic virtual environments. This level of immersion has propelled VR from a specialized niche to a mainstream platform for a range of applications including gaming [16], social interaction [10], remote conferencing [9] and virtual tourism [2]. As VR technologies continue to mature, their adoption rates have skyrocketed. As of February 2023, over 171 million people worldwide, with 65.9 million in the U.S. alone, were using VR applications [1].

While the VR technology enables exciting new applications, it also raises pressing security and privacy concerns. These concerns are not merely hypothetical and could be a significant barrier to VR technology adoption [23]. These issues are further exacerbated by the fact that VR platforms often interface with other smart devices, such as mobile phones and wearables, creating a complex ecosystem ripe for security and privacy vulnerabilities. Existing research efforts in the literature have exposed a variety of security/privacy vulnerabilities in VR platforms, ranging from motion sensor-based inference attacks [82] and eye-tracking exploits that harvest sensitive personal data [41] to exploring how gait and movement data from VR headsets can be used to create deepfake videos [78].

In this paper, we focus on an unexplored, yet highly relevant, privacy risk that arises when a user is simultaneously engaged with a VR platform and a non-VR mobile/wearable device equipped with motion sensors. Individually, an adversary's access to data from either system is usually considered non-threatening. However, within the broader ecosystem, is it possible that a user's privacy may suddenly become vulnerable if an adversary gains access to both data streams? Our work attempts to answer this question by studying if a combination of these disparate data sources can be potentially used to de-anonymize users and compromise their privacy within the VR application ecosystem. Despite recent research efforts focusing on uncovering and overcoming privacy challenges in VR applications, the potential threat (and associated risks) of correlating real-world motion sensor data with in-VR visual data has been largely overlooked. This paper aims to address this specific, yet critical, gap.

This work is the first to systematically investigate the user anonymity and privacy implications stemming from a combination of data from VR and non-VR (mobile/wearable) platforms, with the following specific contributions:

- **Correlation Framework**: We develop a framework for correlating motion sensor data from users' mobile/wearable devices with the visual movement of their virtual avatars in VR applications.
- **Empirical evaluation**: To validate the performance and efficacy of our correlation framework, we collect test data from human subject participants and perform a comprehensive empirical evaluation under various practical settings.
- **Optimization**: We propose improvements and optimizations to our correlation framework, tailored for large-scale attacks. These enhancements provide a more efficient and scalable solution to the identified user-privacy issue.
- **Mitigation strategies**: We discuss potential mitigation policies and recommendations that provide actionable insights for both developers and policy-makers.

2 RELATED WORK

We categorize related research efforts into those that attempt to infer private information from mobile device motion sensors and those that employ VR systems and applications.

Information leakage from mobile device motion sensors: Mobile and wearable device motion sensors such as accelerometers and gyroscopes have been heavily scrutinized in the research literature for their potential to be employed as a side-channel for leaking users' private information. For instance, motion sensor data have been utilized to infer keystrokes and passwords [27, 46, 47, 64], identify lock screen patterns [85], deduce travel routes and location [36, 58, 60], infer speeches [35, 37, 52], infer handwritten text [79], reconstruct 3D models from printer vibrations [67], and estimate demographic information [31, 66]. Application of such on-body motion sensors for user authentication [44, 80, 84] has also received significant attention. However, such biometric authentication systems require training data from individual users.

Information leakage in VR systems and applications: Albeit relatively new as a consumer technology, VR has garnered a host of security and privacy concerns. Attacks such as password inference from finger movements (using motion sensors) when typing a password in the virtual world can become a security problem if the same password is reused by the user in real world [30]. Some VR headsets include eye-tracking, which can reveal valuable personal information [41]. VR, when used in conjunction with Deepfakes [78], can also become a serious threat as an adversary can potentially utilize personal gait and movement data collected from a VR headset to create a very authentic-looking fake video. These type of attacks can be used to damage personal reputation [12], conduct social engineering attacks [81], and spread misinformation [34, 40].

Authentication using authorized sensors on the VR headset or paired on-body controllers [68] and synchronization among multiple (on-body) VR sensors [28, 39] for utility focused applications are another closely related research topics. Unlike these prior research efforts, in our attack we focus on out-of-band motion sensor data, which are not natively paired with the VR system. De-anonymization solely using movements observed in the virtual world is difficult, especially when the confusion set size is large. In this work, we attempt to de-anonymize VR users by correlating visually observed movements of their virtual world avatars with available out-of-band motion data.

One of the most desirable features in a VR experience is the ability to use anonymous avatars and identity transformation. Without appropriate identity protection VR users can be hesitant to participate in the ecosystem [23]. Previous works on de-anonymization of VR users utilized *in-band* data, such as sensors on the VR systems and/or movement characteristics of virtual avatars, to infer users' identity [53, 54], anthropometrics [59], environment [59], device information [59, 72], and demographics [59]. To the best of our knowledge, the proposed de-anonymization attack using out-of-band motion data has never been studied before.

3 SYSTEM AND ADVERSARY MODEL

The target in our proposed attacks are users participating in a VR application. For this, users employ a primary VR device/hardware (from manufacturers such as Microsoft, Meta, Apple and HTC), which typically comprises of a headset running a manufacturer-provided platform or OS. Users are able to execute a variety of manufacturer-provided or third-party VR applications [7, 9–11, 18, 19, 21] on the headset. Users may optionally also employ additional manufacturer-provided or third-party hardware such as hand controllers and headphones/earphones. In addition to the application that the user is directly interacting with, the VR headset may also be running other support applications (in parallel and/or in the background), for example, for live streaming the VR experience.

We refer to all these apps running on the VR headset as *VR-apps*. VR-apps often generate visual/video streams comprising of the users' virtual embodiment (in the form of an avatar) interacting with other virtual users (or avatars) or the virtual environment. Depending on the VR-app, these visual or video stream data may be publicly available to all users of the VR-app or to everyone (in case it is live streamed on a public platform).

While interacting with VR-apps, users may also have in their possession other smart devices such as smartphones and smartwatches. There may be applications running on these smart devices, typically distinct from the VR-apps (e.g., operated by different manufacturers or providers), and are referred by us as *non-VR-apps*. These non-VR-apps may be able to access local sensors on these devices (smartphones and smartwatches) either with or without the explicit permissions of the user. For instance, accessing camera and microphone may require explicit user permissions, while accessing motion sensors such as accelerometers and gyroscopes may not.

We consider an adversary whose goal is to de-anonymize a target VR user (or users) by correlating the visual movements of his/her anonymous virtual world avatar from the publicly-accessible visual/video stream data generated by the VR-app with *out-of-band*, but *identifiable*, mobile/wearable motion sensor data from a set of potential target users collected through non-VR-apps. The *size* of the labeled motion dataset of users (collected from non-VR-apps) in the possession of the adversary, representing the *confusion set* of the target VR user or avatar, may vary between a *large-scale* where the cardinality (of the dataset) may be very high, to a *small-scale*. Similarly, the video recordings of VR users or avatars will result

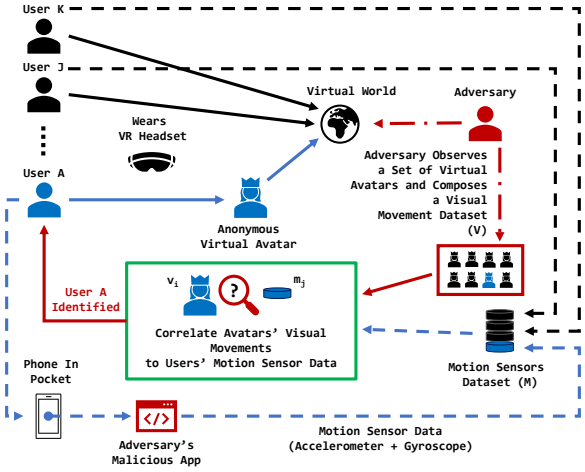


Figure 1: Threat model and Attack Summary

in a visual movement dataset, which can also range between a *large-scale* where its cardinality may be very high, to a significantly smaller *small-scale* such as avatars present within a (targeted) virtual room or playing a (targeted) virtual game. The goal of the adversary, as depicted in Figure 1, is to de-anonymize a target user (i.e., its avatar) in the VR space by matching an element in the labeled motion dataset to the element (corresponding to the target user or avatar) in the visual movement dataset. This adversarial goal can also be extended to include de-anonymization of multiple VR users or avatars.

In order to compile the visual movement dataset (denoted by $V = \{v_1, v_2, \ldots, v_p\}$, with cardinality p), the adversary has to join the virtual world, observe and record each avatar for a baseline duration of time within which a series of movements are likely observed. In case of the VR service provider being the adversary, this process can scale easily. In order to compile the labeled motion dataset (denoted by $M = \{m_1, m_2, \ldots, m_q\}$, with cardinality q), the adversary promiscuously records *zero-permission* motion (accelerometer and gyroscope) sensor data from a targeted set of users. Typically, this can be achieved by means of a malicious SDK and/or a trojan app that offers some utility to the users on the front-end (e.g., a game or a social networking service), while surreptitiously recording the motion data on the back-end. User may also be compelled by a higher authority, for example an employer, school, or government, to download such a (malicious) app onto their smartphone or smartwatch [3, 4, 13, 17, 24]. Several research efforts in the literature have studied/uncovered other significant privacy issues under a similar assumption [27, 45, 48–50, 55, 57, 63, 75–77, 87, 89]. We also assume that both datasets (V and M) contains timestamps which are fairly in sync with the standard global time.

4 CORRELATION FRAMEWORK

Our proposed correlation framework (Figure 2) is composed of two key components. The first component converts both the (out-of-band) motion sensor data and the visual movement data into a comparable format, referred as *activity-vector series*. The activity-vector series enables us to directly compare and match elements from the two datasets (V and M) using a *matching heuristic*. The second component *ranks* the closest matches across the elements of both the datasets, such that the highly ranked matches are most likely associated with the target user (identifiable from M).

4.1 Activity-Vector Series

Our motivation behind defining an *activity-vector series* stems from the fact that the two data sources (V and M) are not directly comparable to each other. The motion sensor data M comprises of samples measuring linear acceleration and orientation changes of a user's body, whereas the visual movement data V consists of video frames recording an anonymous avatar's movements. Consequently, we define an *activity-vector series* as a sequence of activities observed (classified by a ML model), combined with a pairwise sequence of "magnitudes" for each observed activity from each of the data sources. The magnitude quantification (Section 4.5) associated with an observed activity is approximate, but serves as a critical attribute in our correlation framework.

More precisely, our activity-vector series is composed of the following commonly observed activities: *idle*, *body rotation*, *head rotation*, *hand movements*, *walking*, *bending*, *jumping*, and "*other*". These were the common movements observed in over 2000 hours of activity data collected by us inside VRChat [21] (more details in Section 5). These activity classifications combined with magnitude calculations form a vector-like representation where each observed activity has a corresponding magnitude information. An activity-vector series from either data sources can be depicted as:

Left-front Hip Pocket (Motion Sensor)

Activity	*walking*	*walking*	*idle*	*bending*	*walking*	*walking*	*jumping*	*idle*	*walking*	*jumping*
Magnitude	a_4	a_3	a_1	a_7	a_6	a_{10}	a_8	a_2	a_5	a_9

where $a_i \in \mathbb{R}^+$ is the positive real magnitude of an activity time window, such that $a_1 > a_2 > \ldots > a_{10}$. In order to generate this activity-vector series, we next detail the steps taken to pre-process and utilize supervised machine learning models to classify the activities observed in individual sequences.

4.2 Pre-Processing

We first segment both the physical motion data (obtained from the mobile device motion sensors) and the visual movement data (obtained from the VR apps) into small time windows (of w seconds each) and classify each window as one of the eight aforementioned actions. We empirically evaluate the effect of the size of w on correlation accuracy in Section 6.1 and use the optimal value for rest of the evaluation. For the visual movement data, we further separate individual user's avatar from the background, so as to better classify the movements of the avatar without any background noise. *PaddleSeg* [29], an open-source toolkit that applies image segmentation using different techniques, was used to segment out the individual avatars. More specifically, we used a pre-trained *ORCNet* model with *HRNet* backbone that was trained using the *Cityscapes* dataset [15]. For the motion sensor data, we apply a *Savitzky-Golay* filter [61] to smooth the signals for noise reduction before classification.

4.3 Training Data Generation

To generalize and scale our activity classification for a *large-scale* attack, we generate a training dataset by adding synthetically generated variations that capture a wide range of bodily variances

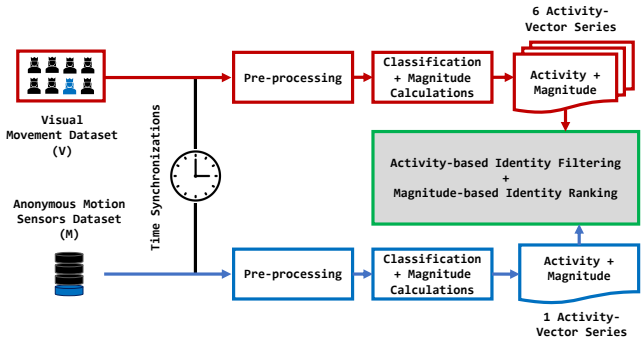


Figure 2: Overview of our correlation framework.

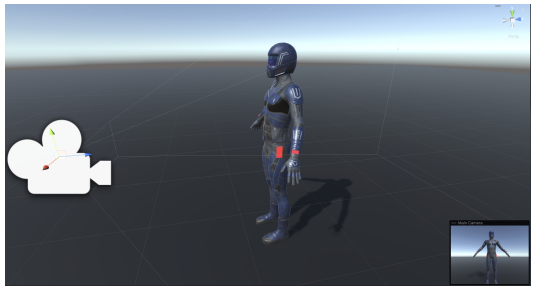


Figure 3: The training data generation setup inside Unity, depicting one camera viewpoint and virtual motion sensors attached to the avatar (in red).

and anomalies (often caused by imperfections in the VR systems), otherwise imfeasible to collect from real human subjects, to a well-known visual movement dataset in the literature. Specifically, we generate the training data of our visual movement classifier using the 3D game engine Unity [20] (Figure 3), utilizing the *CMU MoCap* [5] dataset and synthetically generated variations of motions captured in the CMU MoCap dataset. The CMU MoCap dataset was created using a motion capture system where the subjects wore 41 markers and performed various activities. It is a well-known dataset for evaluation of activity recognition frameworks [26, 56, 65], and can be applied to reproduce avatar movements inside Unity using corresponding body keypoints.

Our synthetically generated movement variations randomized the speed between 0.25× and 2× of CMU MoCap speeds, and rotation angle between −10° and +10° of CMU MoCap rotation angles. In addition to the CMU MoCap model avatar, we also train using another freely available avatar, namely the *Futuristic soldier - Scifi character*[1]. As the video movement data is dependent on the viewpoint of the adversary, we also capture varying camera positions around the virtual avatar in Unity (Figure 3). Specifically, the camera position was randomized around the avatar (across all angles for which the avatar is visible), enabling different visual perspectives and thus improving classifier training. The visual movements of avatars were recorded using OBS Studio [14].

Additionally, in Unity we attached a custom-made *virtual* motion sensor to the avatar (Figure 3), which is able to capture acceleration and orientation changes of the avatar. This virtual motion sensor closely captures the kinematic forces experienced by the avatar in the same way a smartphone or smartwatch motion sensor on a real person would experience, and it allows us to collectively train a classifier for the motion sensor data alongside the visual movement classifier. Such a strategy of using virtual sensors enables training data collection without requiring real human subject participants and also eliminates related synchronization errors. Even though our activity classifiers were trained with publicly-available avatar datasets (and synthetically generated variants of it), for the experimental evaluation and testing of our framework (from an adversarial standpoint) we compose a *test* dataset with the help of real/actual human subject participants and also address synchronization errors between the motion sensor and visual movement data (Section 6). Such a strategy is not only easily generalizable but also realistic, as in practice an adversary may be unable to train activity classifiers using the target users' data.

4.4 Activity Classification

We collectively utilize Apple's Core ML[2] and Create ML[3] libraries to generate two classification models (each trained separately), one using the video movement training data and another using the motion sensor training data as described above. Core ML is a state-of-the-art model pre-trained by Apple for generic action and activity classification, and we further fine-tuned it with the help of transfer learning [51] using the training data described in Section 4.3. Prior research has already demonstrated the feasibility of such activity recognition using Core ML [42]. Moreover, Apple's Vision framework[4] is already pre-trained for keypoint detection on humans, which can also be utilized with Core ML on humanoid avatars. *Applying these trained models on test visual movement and motion sensor data (as described in Section 5.4), split into w second windows, will result in a sequence of activities observed on the two data sources, which is one of the two sequences in the activity-vector series described earlier.*

4.5 Activity Magnitude

Intuitively, when the same classified activity is observed in both data sources (in a given time window), we can improve our identity correlation by *ranking* smaller magnitude differences above larger magnitude difference. For example, if an anonymous avatar is observed to be jumping fast in the virtual world (high magnitude), it is likely that their activity magnitude will also be high on the motion sensor data. As mentioned earlier, our magnitude quantification of an observed activity is approximate. For the motion sensor, we calculate magnitude of each w second activity window as the *average magnitude of acceleration vectors* in the motion sensor data. For the visual movement data, we utilize *optical flow* to compute the average acceleration of areas on the avatar's body where the motion sensor may be attached. Optical flow estimates the motion of objects between consecutive frames in a video, caused by the relative movement between the object and camera [33, 38].

[1] https://assetstore.unity.com/packages/3d/characters/humanoids/sci-fi/futuristic-soldier-scifi-character-202085

[2] https://developer.apple.com/documentation/coreml
[3] https://developer.apple.com/documentation/createml
[4] https://developer.apple.com/documentation/vision

However, as some activities tend to generate disproportionate levels of motion in various parts of the body, it may result in different magnitudes of movements for the same activity. Furthermore, as the adversary may not have knowledge of the motion sensor's positioning for each user's data, the visually observed magnitude of movement experienced by an avatar's different body keypoints is another attribute that should be factored in to improve our correlation model. We consider *six* usual body positions where the motion sensor is likely to be attached, such as a smartphone in pant pocket or a smartwatch on the wrist: left-front hip pocket, right-front hip pocket, left-back hip pocket, right-back hip pocket, left wrist, and right wrist. As a result, the activity-vector series calculated from the visual movement dataset will consists of six different magnitude sequences (for the same activity sequence) as follows:

Left-front Hip (Visual)

Activity	*walking*	*walking*	*idle*	*bending*	*walking*	*walking*	*jumping*	*idle*	*idle*	*jumping*
Magnitude	a_4	a_3	a_1	a_7	a_6	–	a_8	a_2	a_5	a_9

Right-front Hip (Visual)

Activity	*walking*	*walking*	*idle*	*bending*	*walking*	*walking*	*jumping*	*idle*	*idle*	*jumping*
Magnitude	a_4	a_3	a_2	a_7	a_6	a_{10}	a_8	a_1	a_5	a_9

Left-back Hip (Visual)

Activity	*walking*	*walking*	*idle*	*bending*	*walking*	*walking*	*jumping*	*idle*	*idle*	*jumping*
Magnitude	a_4	a_2	a_3	a_7	a_6	a_{10}	a_8	a_1	a_5	a_9

Right-back Hip (Visual)

Activity	*walking*	*walking*	*idle*	*bending*	*walking*	*walking*	*jumping*	*idle*	*idle*	*jumping*
Magnitude	a_4	a_3	a_1	a_7	a_6	a_9	a_8	a_2	a_5	–

Left Wrist (Visual)

Activity	*walking*	*walking*	*idle*	*bending*	*walking*	*walking*	*jumping*	*idle*	*idle*	*jumping*
Magnitude	a_3	a_4	a_1	a_6	a_7	a_8	–	a_2	a_5	a_9

Right Wrist (Visual)

Activity	*walking*	*walking*	*idle*	*bending*	*walking*	*walking*	*jumping*	*idle*	*idle*	*jumping*
Magnitude	a_1	a_3	a_4	a_6	a_7	a_9	a_8	a_2	a_5	a_{10}

where "–" implies unobservable position for optical flow calculations, all a_i in red depict mismatched magnitude rank with the left-front hip pocket motion sensor activity-vector series shown in Section 4.1, and all green a_i imply matching magnitude rank. Moreover, there is an activity misclassification in this example at the ninth window, highlighted as *idle*. All of these seven magnitude sequences (one from motion sensor data and six from visual movement data) are utilized in the correlation and identity ranking processes described next.

4.6 Correlation and Identity Ranking

The first intuitive assumption in our correlation framework is that the order of activities by a user (and his/her avatar) will be unique when observed for a long enough duration. Intuitively, this observation duration can be shorter in a *small-scale* attack where the confusion set is smaller. In a *large-scale* attack, the observation duration has to be longer because with a large confusion set the occurrence of more than one anonymous user conducting the same sequence of activities within a short observation duration is more probable, thus creating confusion between them. We use this assumption to filter out unlikely matches from our identity ranking calculations, using the activity sequences in the activity-vector series.

Our second intuitive assumption is that varying activity magnitudes caused by disproportional levels of motion in various parts of the body can be utilized to identify closely correlated visual movement and motion sensor sequences. Accordingly, we utilize magnitude correlation rankings to rank known identities (from dataset M) such that users with motion sensor magnitude sequence closely matching to a visual movement magnitude sequence (best of the six visual positions) are ranked closer to 1.

4.6.1 Activity-based Filtering. As the activity classification is not perfect, we cannot reliably use the sequence of activities for correlation. Instead, we use a high degree of mismatch between sequences of activities (across visual movement and motion sensor data) to filter out identities whose motion sensor data are objectively different from an anonymous avatar being observed. More specifically, we calculate the *Hamming distance* between the motion sensor activity sequence and the visual movement activity sequence (which is the same for all six activity-vector series generated from the visual movement data). Thereafter, we eliminate pairs with distance threshold $> t$ from further magnitude-based identity rankings. We empirically evaluate threshold t in Section 6.1 as part of our framework parameter optimization. For example, between the pair of activity-vector series illustrated in Section 4.1 and Section 4.5, this Hamming distance is 1 (or 10%) due to the activity mismatch in the ninth time window.

4.6.2 Magnitude-based Ranking. After filtering, we are left with identities whose motion sensor activity sequences closely matched at least one of the six visual movement activity sequences. We utilize *Spearman's rank correlation coefficient* [88] to correlate and rank potential identities based on magnitude sequences, which is computed as follows:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

where n is the number of observations (of w second windows) in the activity-vector series and d_i is the difference in the paired ranks of the two magnitudes (across the visual movement and motion sensor data sequences) at the i^{th} time window. The higher the Spearman's coefficient, the more likely the two sequences correlate to each other, and thus the corresponding identity from M would be ranked closer to 1 out of the q (minus the identities that did not pass the activity-based filtering). As the adversary does not have positioning information of the motion sensor on the users' body, we compute Spearman's correlation coefficient for the six likely positioning of the motion sensors (Section 4.5), and consider only the maximum for identity ranking. Between the examples shown in Section 4.1 and Section 4.5, magnitude from the visual data sequence of the left-front hip will have the highest Spearman's correlation coefficient with the left-front hip pocket motion sensor magnitudes. When activity-based filtering threshold t is set very low (i.e., only tolerance for very minor or no mismatches in the activity sequences), it is also possible that all identities are eliminated from this magnitude-based raking, thus resulting in no identity ranking. The entire correlation procedure is described in Algorithm 1.

Algorithm 1 Correlation Algorithm.

Input:
  $video[]$ ▷ Video's activity-vectors series
  $motion[]$ ▷ Motion's activity-vectors series
  t ▷ Filtering threshold
Output:
  $ranked[]$ ▷ Ranked list of correlated motion-video indexes with maximum Spearman's RCC
procedure CORRELATE
  $correlated[]$ ▷ Maps motion indexes to correlated video indexes
  $unranked[]$ ▷ Unranked list of correlated motion-video indexes with maximum Spearman's RCC
  for i in range($video.size() - 1$) **do**
    for j in range($motion.size() - 1$) **do**
      if $HammingDistance(video[i], motion[j]) < t$ **then**
        $correlated[i].append(j)$
      end if
    end for
  end for
  for i in range($video.size() - 1$) **do**
    for j in range($correlated[i].size() - 1$) **do**
      $m_{idx} = correlated[i][j]$ ▷ Motion index
      $maxSpearman = max(Spearman(video[i], motion[m_{idx}]))$
      $unranked[i].append(\{maxSpearman, m_{idx}\})$
    end for
    $ranked[i] = unranked[i].sort()$ ▷ Sorted based on Spearman's RCC
  end for
end procedure

5 EXPERIMENTAL SETUP

To evaluate our proposed correlation framework and training methodology, we collect *test data* (visual and motion sensor) from human subject participants using a real VR application. Here we outline details of our test data collection procedure.

5.1 Participants' Task

Our participants (details in Section 5.3) carry out a set of representative activities in a virtual reality app while carrying a smartphone and smartwatch on their body. The *controlled* activities include movement of the head, arms, palms, legs, and also actions that require combinations of them. These different actions were chosen to generate a wide variety of different movements within the limited time we had with the participants. During the *uncontrolled* activity phases, participants were free to interact with the VR app on their own volition, not limited by the aforementioned activities. The average time each of our participants spent on the VR app to provide us data for our study was 1 hours and 8 minutes.

5.2 Adversarial Viewpoint

We continuously observe and record the participants' avatar in the virtual world by means of five different virtual camera positions, where each camera position represents a different adversarial viewpoint. Four of these positions are static and positioned at different corners of the virtual room, each of which represents the fixed (or static) position of an adversarial avatar observing the target participant from that position. The fifth camera is mobile, and represents the view of an adversarial avatar moving and navigating in the proximity of the (target) participant's avatar. We carried out our experiments in two different virtual worlds – one in a public world (called *Black Cat*) where other users' avatars may be present, and second in a private world (called *Home*) where access is restricted to a select group of users. We refer to these five adversarial viewpoints in these two worlds by means of a legend outlined in Table 1. In our evaluation (Section 6), we will also analyze the effect of combining these five viewpoints on the accuracy of activity classification (where the viewpoints are referred to as HCC and BCC for Home and Black Cat, respectively).

Table 1: Legend of camera viewpoints used in Section 6.

Home	Legend	Black Cat	Legend
Static Camera 1	HC1	Static Camera 1	BC1
Static Camera 2	HC2	Static Camera 2	BC2
Static Camera 3	HC3	Static Camera 3	BC3
Static Camera 4	HC4	Static Camera 4	BC4
Mobile Camera	HC5	Mobile Camera	BC5
Combined	HCC	Combined	BCC

Table 2: Background details of the 35 participants.

Gender	
14 Female	21 Male
Dominant Hand	
2 Left	33 Right
VR Familiarity	
11 Slightly	24 Moderately-Extremely
Prior VR Experience	
5 Never Used VR Before	30 Used VR Before

5.3 Participants

We recruited 64 participants for test data collection, however, due to various personal, technical, and medical challenges, only 35 of them completed the study and whose data is included in our evaluation. The participants' ages were between 18 and 48, with a median age of 19. Additional demographic and other details about our participants are listed in Table 2. Participants were appropriately compensated for their time and our study was approved by our institution's Institutional Review Board (IRB).

5.4 Data Collection Apparatus

VR Device and App. We utilize the Meta Quest 2 VR device[5] and the popular VRChat [21] app (installed on the Quest 2) for generating and collecting test data from the participants in our study. As of July 2022, VRChat had more than 200,000 daily active users and more than 7 million registered users [22]. Although other popular apps also have full-body avatars [8], the fundamental nature of data generation (and collection) does not significantly differ across a majority of the VR apps.

Motion Sensors. Participants' body motion was captured at 20 *ms* sampling interval on a smartwatch (TicWatch 2) worn by the participants on their wrist and on a smartphone (Moto G7 Play) placed in their pocket. 10 participants chose to wear the smartwatch on their right wrist, while the rest chose to wear it on their left wrist. 23 participants placed the smartphone in one of their front pockets, while the rest place it in one of their back pockets.

Data Logging. The VRChat app was installed on five different desktops to record the viewpoints/perspective of an adversary as described in Section 3, and OBS Studio [14] was used to record the each adversarial perspective into individual video files with timestamps. The motion sensors were logged in respective devices with timestamps, and later transferred to another desktop for analysis.

Analysis Computer. A MacBook Pro, equipped with 10-Core M1 CPU, 16-Core GPU, 16GB memory, 1TB SSD storage and 16-core Neural Engine, was used to train and classify activities, and for the activity-based filtering and magnitude-based ranking tasks. For the

[5]https://www.meta.com/quest/products/quest-2

large-scale analysis (Section 7), we used a desktop with Ryzen 5 3600 6-Core 3.6GHz CPU, RTX 3060 12GB GPU, 1TB SSD storage, and 16GB memory, to train and generate large datasets using CTGAN [6, 83].

6 EVALUATION

We evaluate the proposed correlation framework utilizing the test data collected from participants, which represents a *small-scale* attack with confusion set size of 271 (accumulating different motion sensor locations from individual participants). After comprehensively evaluating the framework in the *small-scale* setting, we generate and evaluate a representative dataset for a *large-scale* correlation in Section 7.

6.1 Framework Parameters

Our correlation framework has two key parameters: (i) activity window size (w), which is the time duration used to classify an action, and (ii) Hamming distance used as the activity-based filtering threshold (t), which is the minimum requirement for an activity-vector to be considered in the identity ranking. As the total observation time, and thus the number of observed activity windows, will vary between different target users, the activity-based filtering threshold (t) is normalized with respect to the number of observed activity windows. No filtering occurs when the filtering threshold is set at 100%, whereas at 0% even one mismatch in the activity sequence will result in that activity-vector being filtered out.

Figures 4 and 5 show the correlation accuracy, where "None Correlated" occurs when the activity-based filtering filters all candidate activity-vectors, "Incorrectly Correlated" occurs when the top ranked identity is incorrect, and "Correctly Correlated" occurs when the top ranked identity is correct. From these figures, we can see that as we increase w, the percentage of identities that passes the activity-based filtering and then used for identity ranking also grows. Conversely, the percentage of "None Correlated" is diminished as w is increased. This can primarily be attributed to: (i) the size of activity sequence in the activity-vector is inversely proportional to w for a constant observation time period thereby reducing the number of probable mismatches, and (ii) the activity inference tends to perform more accurately for larger w.

While the above observation should compel us to select a larger w, in Figures 4 and 5 we also observe that there exists a trade-off between w and correctly correlated identities for different activity-based filtering thresholds. For instance, when $w = 5s$ we observe that the percentage of correctly correlated identities starts to decrease beyond the filtering threshold of 70% in Figure 4e. This is most likely because as the size of activity-vector is reduced with increasing w, the probability of confusion with another person's activity magnitudes is increased. This trend was consistent across other experimental variables, such as different adversarial viewpoints, motion sensors, and motion sensor positions on the body.

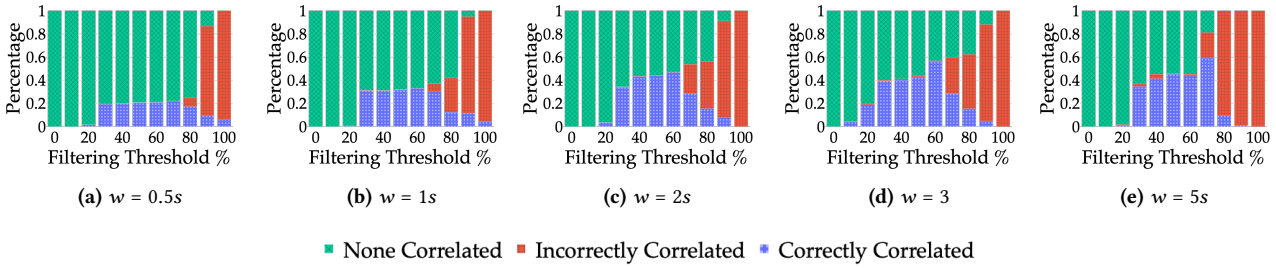
Based on empirical observations across different experimental variables, we set $w = 1s$ and $t = 30\%$ for the rest of our analyses. On average, these selected values are best suited for maximizing the percentage of correctly correlated identities. The average correctly correlated identities using these parameter values within *top-1* of the ranking was 16.3%, and 17.0% of the identities were within top-3. These values are significant as training did not consider participant data at all! In an alternate adversarial model where the motion sensor positions on the body is known to the adversary, more specific (i.e., per target user) w and t values can be selected to further improve the percentage of correctly correlated identities.

6.2 Activity Confusions

The accuracy of the activity classification models play an important role in the correlation framework's overall success rate. Activity classification between visual and motion sensor data differs significantly due to the modality of input signal, and is subject to different types of noises and interference signals. Different adversarial viewpoint angles, distances, and occlusion levels affect the visual data classification. For instance, if only half of the avatar is visible due to being behind a coach or another avatar is in front of the target avatar, the chance of a misclassification is significantly increased. The positioning and orientation of the device used to collect motion sensor data also imposes certain limitations on the activity classification accuracy, especially as we assume that the adversary is unaware of the exact position of the motion sensor. For instance, if the motion sensor data is from a smartwatch worn on the right hand, it is very useful to classify activities involving the right hand, but may result in high misclassification of activities not involving the right arm.

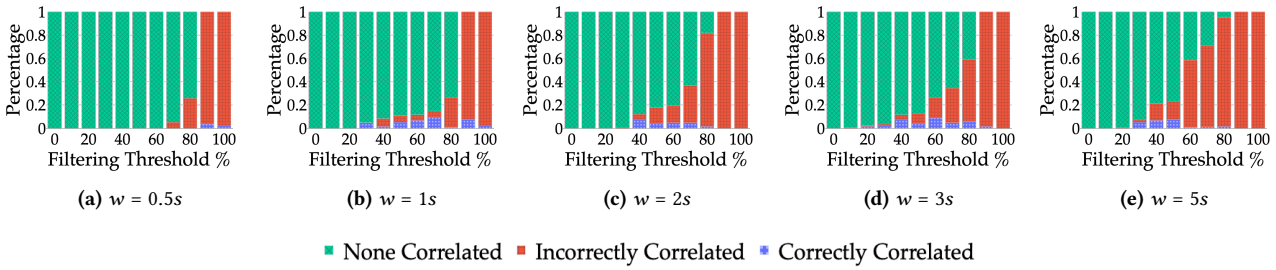
Due to these apparent limitations, we analyze the direct consequence of misclassifications, i. e., the confusion of activities between the visual and motion sensor data. In Figure 6, we observe that the *idle* activity has noticeably low accuracy (36% and 22% for right wrist smartwatch and front right pocket smartphone, respectively), and is often confused with other activities. An unexpected, yet clearly discernible, confusion exists between motion sensor *idle* and visual *walking*. One possible factor behind this observation is that VR users may be using the VR joystick to walk in the virtual world. As a result, the target user appears idle in the motion sensor data, while their virtual avatar is visually walking. Another noteworthy observation is that head movements had high confusion due to the fact that placement of motion sensors around hip and wrist areas is not suitable for capturing the target user's head movements, whereas a head-mounted VR device is accurately able to capture head movements and apply them to the avatar in the virtual world.

In light of these insights, we further optimize our framework as follows. Rather than considering all the classified actions, we only utilize activities with less than 60% of confusion – body, hand, walk, bend, jump, and others – for our activity-based filtering. Remaining activities in the activity-vector are ignored from the Hamming distance calculations. The average correctly correlated identities after this optimization within top-1 of the ranking was 37.3%, while 38.7% of the identities were within top-3. The correlation accuracy plateaued beyond top-3 due instances of the real identity's activity-vector series being eliminated from the rankings by the activity-based filter. This suggests that as activity classification models improve in the future, our attack's correlation accuracy will also improve.



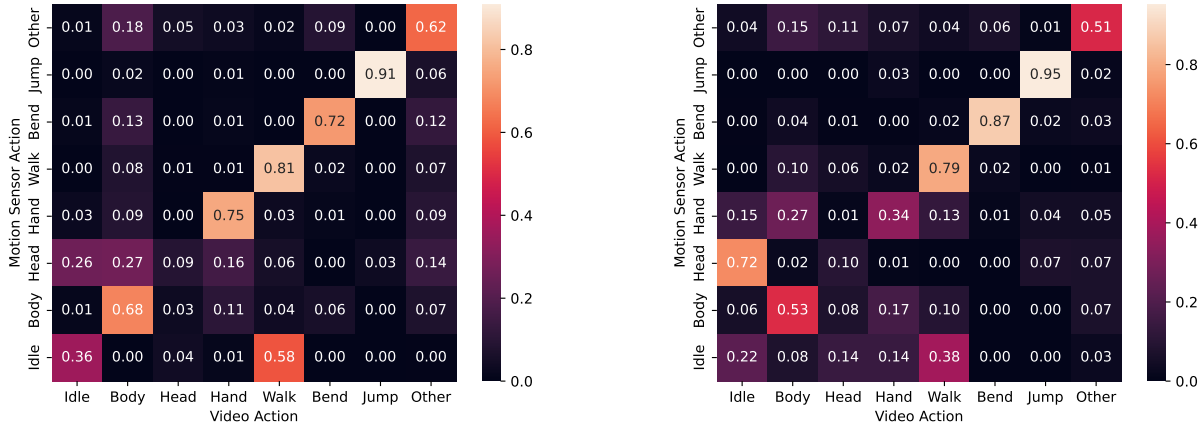
(a) $w = 0.5s$ (b) $w = 1s$ (c) $w = 2s$ (d) $w = 3$ (e) $w = 5s$

Figure 4: Right smartwatch motion sensor and visual movement data correlated with different w and normalized t parameters. Accuracy based on top-1 identity in the rankings.



(a) $w = 0.5s$ (b) $w = 1s$ (c) $w = 2s$ (d) $w = 3s$ (e) $w = 5s$

Figure 5: Front right pocket smartphone motion sensor and visual movement data correlated with different w and normalized t parameters. Accuracy based on top-1 identity in the rankings.



(a) Using right wrist smartwatch. (b) Using front right pocket smartphone.

Figure 6: Activity classification confusion between motion sensor data and visual movements.

6.3 Time Alignment

Both the visual and motion sensor data are collected with device timestamps for synchronization. Although most modern mobile devices are periodically updated using Internet time servers, motion sensor data collection in the wild may contain time drift errors and thus misaligned with the visual movements. Misaligned data sources will likely cause confusion between classified activities, resulting in a high failure rate in satisfying the activity-based filter threshold. As shown in Figure 7, misaligned data can drop a 62.1% correctly correlated result down to 0% in the presence of only 2.4 seconds (of artificially introduced) misalignment. The adversary can potentially detect and overcome such misalignments by offsetting the (motion sensor) data in increments, and selecting a time offset ($\pm\delta$) that results in the minimum Hamming distance in the activity-based filtering. The value of δ must be appropriately chosen to keep the computation time practical.

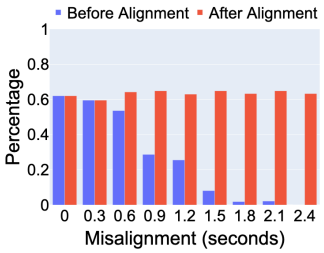


Figure 7: Correctly correlated accuracy (top-1 rank) with artificially introduced misalignment, shown for data from the right wrist.

6.4 Different Sensor Locations

We next detail how different positions of the motion sensor on the (human) body and different adversarial viewpoints affect the correct correlation of our proposed framework. Overall, smartwatch (motion sensor) on left or right wrist performed better than the smartphone in the hip pockets (Figure 8). For example, for the *Home* world the smartwatch yielded about 41% and 68% correct correlations (top-1 rank), for left and right wrists, respectively. In contrast, the front left-front pocket smartphone data resulted in about 9.1% correct correlations, while other smartphone locations are in a similar range. Intuitively, one of the main factors behind this observation is the inability of smartphone motion sensors to pick up hand movements when they are located in the hip area pockets. This causes higher confusion between activities (Figure 6), resulting in the activity-vector of the target user being filtered out with high likelihood. As far as the impact of different adversarial viewpoints on the correlation accuracy of our framework is concerned, we can see from Figure 8 that, except for BC1, all other camera locations (or adversarial viewpoints) yielded comparable results within each of the motion sensor locations. The reason behind BC1 performing particularly poorly is that its location was near the entrance point of the *Black Cat* world and most participants eventually moved away from the field-of-view of this camera during the data collection experiments. In summary, combining multiple viewpoints and the availability of wrist-based motion sensor data are the most favorable conditions for the adversary.

6.5 Similar Activity Sequences

There can be situations where multiple users perform a similar or even an identical sequence of activities. In such cases, the magnitude-based ranking should ideally still rank the real identity (of the target user) higher than others. In this part of our analysis, we study the extent to which our magnitude-based ranking is able to do so, by comparing correlation accuracy when participants (and their avatars) performed the same sequence of activities. In Figure 9b, we observe 16.5% correct correlation for motion data from the right wrist in top-1 of identity rankings and 50.1% correct correlation within the top-3 ranks. This demonstrates that magnitude-based ranking is able to, to an extent, discern the difference between identities based on the magnitude of movements.

7 OPTIMIZING FOR LARGE-SCALE ATTACKS

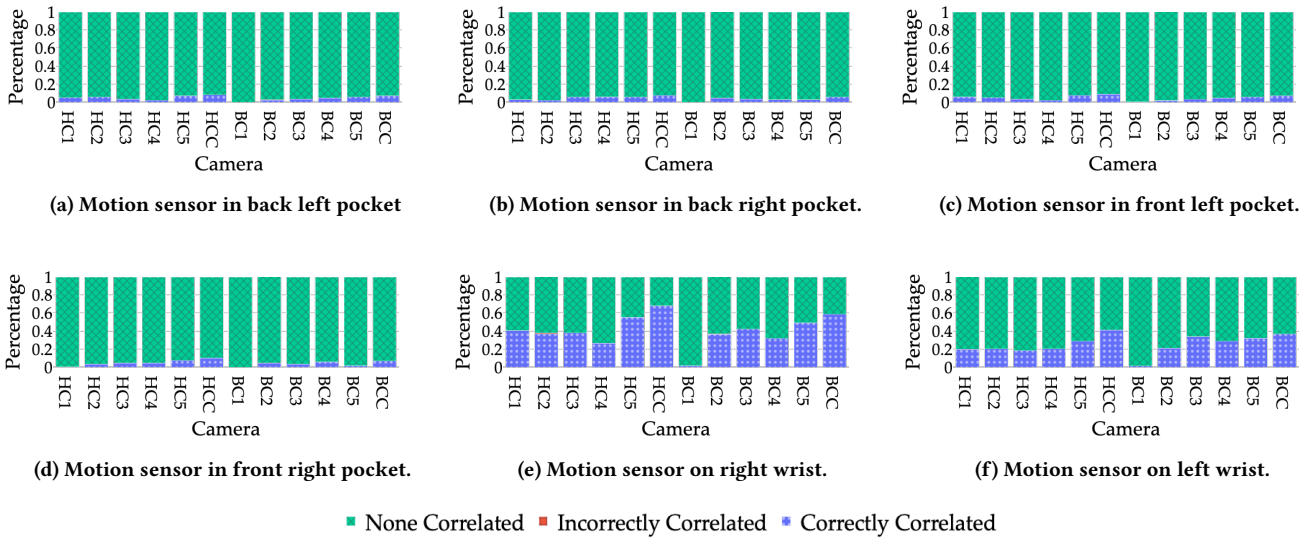
An adversary trying to correlate thousands (or even millions) of anonymous avatars with identified motion sensors data is presented with a significant computational task. As there are various ongoing research efforts on improving human activity classification performance [25, 32, 43, 62, 69–71, 73, 74, 86] which can improve our attack accuracy, in this section, we only focus on the computational complexity of the correlation task and propose related optimizations to our framework.

Synthetic Data Generation To test the scalability of our framework, we must first generate a very large synthetic dataset *utilizing* real participant data collected in Section 5. While it was not feasible for us to collect real-world data from a very large number of participants, due to the time and resources required for systematic data collection per participant, we still want to test using a dataset that has some resemblance to the *small-scale* dataset instead of generating random activity-vectors. Nonetheless, as we are only analyzing computational complexity of a *large-scale* attack, the realism and diversity of our synthetic dataset, and corresponding correlation accuracy results, is not a significant concern.

The activity classification and magnitude calculation tasks take constant time, and will grow linearly with the size of each dataset (p and q, for visual movement and motion sensor datasets, respectively). For large p and q, the more complex task is that of calculating the correlation of all q identities against all p anonymous avatars. However, as seen in Section 6, the activity-based filtering is very effective in reducing the complexity of the magnitude-based identity rankings. Therefore, for large p and q the most computationally complex task in the entire framework comes down to the activity-based filtering. Accordingly, we generate our *large-scale* dataset to test the scalability of our activity-based filtering, which only requires activity sequences as input. Our first *large-scale* dataset was generated using a modern tabular Generative Adversarial Network (GAN) technique [6], called CTGAN [83], which is trained using activity sequences from real participants, as outlined in Section 5. Our second *large-scale* dataset was generated using random permutations of our activity sequences from Section 5. Each of these *large-scale* datasets contained 1 million activity sequences for both the motion sensor and avatar visual movement data.

Activity-based Filtering without Optimizations. Without any optimizations, the activity-based filtering has a time complexity of $O(pqk^2)$, where p is the number of unique avatars from the visual movement data, q is the number of different identities from the motion sensor data, and k is the size of the activity sequences. As such, we can further assume that increasing the size of k would have diminishing returns (computationally), making it less attractive for an adversary to record each target for too long. Therefore, we assume k would not be scaled, unlike p and q, and treat k as constant, thus resulting with a complexity of $O(pq)$. For instance, our setup takes 2.2×10^1 ms to finish activity-based filtering when $p = q = 100$. However, when we scale up to $p = q = 10^5$, it requires 3.15×10^7 ms (or about 8 hours). We estimate that for $p = q = 10^6$, it will take approximately 30 days to finish, and about 3000 days when $p = q = 10^7$, which is not very scalable.

Optimization. We propose the use of a hash table to store our activity sequence data in order to reduce the time complexity of



(a) Motion sensor in back left pocket

(b) Motion sensor in back right pocket.

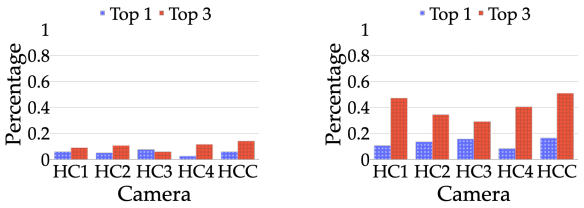
(c) Motion sensor in front left pocket.

(d) Motion sensor in front right pocket.

(e) Motion sensor on right wrist.

(f) Motion sensor on left wrist.

Figure 8: Accuracy for different cameras positions and motion sensors locations of devices during the free-movement phase. Accuracy based on top-1 identity in the rankings.



(a) Front left pocket motion data

(b) Right wrist motion data

Figure 9: Identity correlation for similar activities.

activity matching and filtering. However, as even a single mismatch between two activity sequences will result in completely different hash values (i.e., the keys in a hash table), we design a larger hash table that allows for some degree of mismatch. Specifically, we populate a hash table with keys based on permutations of the q activity sequences in M (each of length k) from the motion sensors data, accounting for possible errors allowable within the Hamming distance threshold (t). Let us assume that the numbers 0 to 7 denotes each of the eight activities we classify. If $k = 5$, an example of the activity string would be $\langle 47634 \rangle$. If our hamming distance threshold is $t = 2$, then any two activities can be mismatched and still pass the threshold. Now, assume the character $*$ as a wildcard activity that may or may not be a match. To populate the hash table exhaustively, we compute every possible permutation of each activity sequence in M including up to two $*$. For our previous example, $\langle 47634 \rangle$, some of the permutations generated would be $\langle * * 634 \rangle$, $\langle 4 * 6 * 4 \rangle$, and $\langle 47 * 3 * \rangle$. All these permutations are then used as the key in our hash table, while the corresponding value is the identity of users from the motion sensor data (M). Thereafter, during the correlation process, each activity sequence from the video dataset also undergoes permutations with up to two $*$, and then queried against the above hash table for a match. If a matching key exists, the corresponding identity and activity-vector has satisfied the activity-based filtering and is included in the identity ranking.

Optimized Performance Analysis. The number of permutations per activity-vector does not scale with the size of datasets and thus can be treated as $O(1)$ time complexity. Similarly, hash table search and insertion is $O(1)$ time complexity. Therefore, with the use of our hash table, the new time complexity becomes $O(p + q)$, where $O(q)$ time is required to create the hash table, and $O(p)$ time is require to iterate through V for filtering. Our empirical results show that with this optimization, the activity-filtering is significantly faster. For instance with $p = q = 100000$, $k = 10$, and $t = 3$, using the optimization technique was 575 times faster than the default activity-based filter. Another important aspect we also evaluate in the empirical results is the number of collisions, which occur when multiple unique data sources satisfies the activity-matching threshold. For $k = 5$, we observe 1594.87 average number of collisions, and for $k = 10$, we observe only 4.26 average collisions (with $p = q = 100000$). This implies that the adversary should increase the value of k if it observes a very high number of collisions.

8 CONCLUSION

We proposed a novel framework to correlate anonymous avatars in virtual worlds with identified out-of-band motion sensor data. Our work highlights a newfound privacy risk to users of the growing VR ecosystem. Specifically, VR users can be vulnerable to de-anonymization attack if they carry a smartphone or wear a smartwatch while using a VR system. Our evaluation of the proposed framework is a step towards demonstrating the feasibility of such an attack, utilizing real-world data from human participants. Through our empirical analyses, we were able to optimize framework parameters, improve scalability, and identified current limitations and potential for further improvements.

REFERENCES

[1] Online; accessed 2023. 25+ Amazing Virtual Reality Statistics. https://www.zippia.com/advice/virtual-reality-statistics.

[2] Online; accessed 2023. 7 Great Virtual Reality Travel Experiences. https://www.lifewire.com/virtual-reality-tourism-4129394.

[3] Online; accessed 2023. Can an employer force you to download apps for work on your personal phones? https://www.reddit.com/r/LegalAdviceUK/comments/n0em70/can_an_employer_force_you_to_download_apps_for/.

[4] Online; accessed 2023. Churches are demanding members download 'invasive spyware' to check if they are watching porn. https://flipboard.com/topic/conormcgregor/churches-are-demanding-members-download-apps-that-spy-on-their-activity/a-IQ2TlaPMRcKAP4RIzkSmtg%3Aa%3A114859074-692d111f36%2Fthe-sun.com.

[5] Online; accessed 2023. CMU Graphics Lab Motion Capture Database. http://mocap.cs.cmu.edu.

[6] Online; accessed 2023. GAN-for-tabular-data. https://github.com/Diyago/GAN-for-tabular-data.

[7] Online; accessed 2023. Google Blocks. https://arvr.google.com/blocks.

[8] Online; accessed 2023. Legs are finally coming to Mark Zuckerberg's metaverse. https://www.vox.com/recode/2022/10/11/23399439/metaverse-mark-zuckerberg-connect-avatar-legs-meta-microsoft-apple-vr-ar.

[9] Online; accessed 2023. MeetinVR. https://www.meetinvr.com.

[10] Online; accessed 2023. Meta Horizon Worlds. https://www.meta.com/experiences/2532035600194083/.

[11] Online; accessed 2023. Mozilla Hubs. https://hubs.mozilla.com.

[12] Online; accessed 2023. MrDeepFakes. https://mrdeepfakes.com.

[13] Online; accessed 2023. My school made us download an app and the app tracks us. Is this legal? If not, what do I do and how do I confront them? https://www.quora.com/My-school-made-us-download-an-app-and-the-app-tracks-us-Is-this-legal-If-not-what-do-I-do-and-how-do-I-confront-them.

[14] Online; accessed 2023. OBS Studio. https://obsproject.com.

[15] Online; accessed 2023. ocrnet-hrnet-w48-paddle. https://docs.openvino.ai/latest/omz_models_model_ocrnet_hrnet_w48_paddle.html.

[16] Online; accessed 2023. Oculus Store. https://www.oculus.com/experiences/quest/.

[17] Online; accessed 2023. Should you make your employees download an app? https://www.teamsense.com/blog/should-you-make-your-employees-download-an-app.

[18] Online; accessed 2023. Sketchfab Virtual Reality. https://sketchfab.com/virtual-reality.

[19] Online; accessed 2023. Spatial. https://www.spatial.io.

[20] Online; accessed 2023. Unity Real-Time Development Platform. https://unity.com.

[21] Online; accessed 2023. VRChat. https://hello.vrchat.com/.

[22] Online; accessed 2023. VRChat's estimated total players. https://mmostats.com/game/vrchat.

[23] Devon Adams, Alseny Bah, Catherine Barwulor, Nureli Musaby, Kadeem Pitkin, and Elissa M Redmiles. 2018. Ethics emerging: the story of privacy and security perceptions in virtual reality. In *ACM SOUPS*.

[24] Alex Akinbi, Mark Forshaw, and Victoria Blinkhorn. 2021. Contact tracing apps for the COVID-19 pandemic: a systematic literature review of challenges and future directions for neo-liberal societies. *Health Information Science and Systems* (2021).

[25] Yair A Andrade-Ambriz, Sergio Ledesma, Mario-Alberto Ibarra-Manzano, Marvella I Oros-Flores, and Dora-Luz Almanza-Ojeda. 2022. Human activity recognition using temporal convolutional neural network architecture. *Expert Systems with Applications* (2022).

[26] Mathieu Barnachon, Saïda Bouakaz, Boubakeur Boufama, and Erwan Guillou. 2014. Ongoing human action recognition with motion capture. *Pattern Recognition* 47 (2014).

[27] Liang Cai and Hao Chen. 2011. {TouchLogger}: Inferring Keystrokes on Touch Screen from Smartphone Motion. In *USENIX HotSec*.

[28] Girija Chetty and Matthew White. 2016. Body sensor networks for human activity recognition. In *IEEE SPIN*.

[29] PaddlePaddle Contributors. Online; accessed 2023. PaddleSeg, End-to-end image segmentation kit based on PaddlePaddle. https://github.com/PaddlePaddle/PaddleSeg.

[30] Anupam Das, Joseph Bonneau, Matthew Caesar, Nikita Borisov, and XiaoFeng Wang. 2014. The tangled web of password reuse. In *NDSS*.

[31] Erhan Davarci, Betul Soysal, Imran Erguler, Sabri Orhun Aydin, Onur Dincer, and Emin Anarim. 2017. Age group detection using smartphone motion sensors. In *IEEE EUSIPCO*.

[32] Ishan Dave, Zacchaeus Scheffer, Akash Kumar, Sarah Shiraz, Yogesh Singh Rawat, and Mubarak Shah. 2022. Gabriellav2: Towards better generalization in surveillance videos for action detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*.

[33] Douglas DeCarlo and Dimitris Metaxas. 1996. The integration of optical flow and deformable models with applications to human face shape and motion estimation. In *IEEE CVPR*.

[34] Nicholas Diakopoulos and Deborah Johnson. 2021. Anticipating and addressing the ethical implications of deepfakes in the context of elections. *New Media & Society* (2021).

[35] Jun Han, Albert Jin Chung, and Patrick Tague. 2017. Pitchln: eavesdropping via intelligible speech reconstruction using non-acoustic sensor fusion. In *ACM/IEEE IPSN*.

[36] Jun Han, Emmanuel Owusu, Le T Nguyen, Adrian Perrig, and Joy Zhang. 2012. Accomplice: Location inference using accelerometers on smartphones. In *IEEE COMSNETS*.

[37] Duncan Hodges and Oliver Buckley. 2018. Reconstructing what you said: Text inference using smartphone motion. *IEEE Transactions on Mobile Computing* (2018).

[38] Berthold KP Horn and Brian G Schunck. 1981. Determining optical flow. *Artificial intelligence* (1981).

[39] Daesung Jang, Joon-Seok Kim, Ki-Joune Li, and Chi-Hyun Joo. 2011. Overlapping and synchronizing two worlds. In *ACM GIS*.

[40] Stamatis Karnouskos. 2020. Artificial intelligence in digital media: The era of deepfakes. *IEEE Transactions on Technology and Society* (2020).

[41] Jacob Leon Kröger, Otto Hans-Martin Lutz, and Florian Müller. 2019. What does your gaze reveal about you? On the privacy implications of eye tracking. In *IFIP International Summer School on Privacy and Identity Management*.

[42] Amit Kumar, Kristina Yordanova, Thomas Kirste, and Mohit Kumar. 2018. Combining off-the-shelf image classifiers with transfer learning for activity recognition. In *Proceedings of the 5th international Workshop on Sensor-based Activity Recognition and Interaction*.

[43] Viet-Tuan Le, Kiet Tran-Trung, and Vinh Truong Hoang. 2022. A comprehensive review of recent deep learning techniques for human activity recognition. *Computational Intelligence and Neuroscience* (2022).

[44] Gen Li and Hiroyuki Sato. 2020. Handwritten signature authentication using smartwatch motion sensors. In *IEEE COMPSAC*.

[45] Yang Liu and Zhenjiang Li. 2019. aleak: Context-free side-channel from your smart watch leaks your typing privacy. *IEEE Transactions on Mobile Computing* (2019).

[46] Chris Xiaoxuan Lu, Bowen Du, Hongkai Wen, Sen Wang, Andrew Markham, Ivan Martinovic, Yiran Shen, and Niki Trigoni. 2018. Snoopy: Sniffing your smartwatch passwords via deep sequence learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* (2018).

[47] Anindya Maiti, Oscar Armbruster, Murtuza Jadliwala, and Jibo He. 2016. Smartwatch-based keystroke inference attacks and context-aware protection mechanisms. In *ACM ASIACCS*.

[48] Anindya Maiti, Ryan Heard, Mohd Sabra, and Murtuza Jadliwala. 2018. Towards inferring mechanical lock combinations using wrist-wearables as a side-channel. In *ACM WiSec*.

[49] Anindya Maiti, Murtuza Jadliwala, Jibo He, and Igor Bilogrevic. 2018. Side-channel inference attacks on mobile keypads using smartwatches. *IEEE Transactions on Mobile Computing* (2018).

[50] Philip Marquardt, Arunabh Verma, Henry Carter, and Patrick Traynor. 2011. (sp) iphone: Decoding vibrations from nearby keyboards using mobile phone accelerometers. In *ACM CCS*.

[51] Oge Marques. 2020. Machine Learning with Core ML. In *Image Processing and Computer Vision in iOS*.

[52] Yan Michalevsky, Dan Boneh, and Gabi Nakibly. 2014. Gyrophone: Recognizing speech from gyroscope signals. In *USENIX Security Symposium*.

[53] Mark Roman Miller, Fernanda Herrera, Hanseul Jun, James A Landay, and Jeremy N Bailenson. 2020. Personal identifiability of user tracking data during observation of 360-degree VR video. *Scientific Reports* (2020).

[54] Robert Miller, Natasha Kholgade Banerjee, and Sean Banerjee. 2022. Combining Real-World Constraints on User Behavior with Deep Neural Networks for Virtual Reality (VR) Biometrics. In *IEEE VR*.

[55] Emiliano Miluzzo, Alexander Varshavsky, Suhrid Balakrishnan, and Romit Roy Choudhury. 2012. Tapprints: your finger taps have fingerprints. In *ACM MobiSys*.

[56] Clinton Mo, Kun Hu, Shaohui Mei, Zebin Chen, and Zhiyong Wang. 2021. Keyframe extraction from motion capture sequences with graph based deep reinforcement learning. In *ACM Multimedia*.

[57] Reham Mohamed, Habiba Farrukh, Yidong Lu, He Wang, and Z Berkay Celik. 2023. iStelan: Disclosing Sensitive User Information by Mobile Magnetometer from Finger Touches. *Proceedings on Privacy Enhancing Technologies* (2023).

[58] Arsalan Mosenia, Xiaoliang Dai, Prateek Mittal, and Niraj K Jha. 2017. Pinme: Tracking a smartphone user around the world. *IEEE Transactions on Multi-Scale Computing Systems* (2017).

[59] Vivek Nair, Gonzalo Munilla Garrido, and Dawn Song. 2022. Exploring the Unprecedented Privacy Risks of the Metaverse. *arXiv preprint arXiv:2207.13176* (2022).

[60] Sashank Narain, Triet D Vo-Huu, Kenneth Block, and Guevara Noubir. 2016. Inferring user routes and locations using zero-permission mobile sensors. In *IEEE S&P*.

[61] William H Press and Saul A Teukolsky. 1990. Savitzky-Golay smoothing filters. *Computers in Physics* (1990).
[62] Sen Qiu, Hongkai Zhao, Nan Jiang, Zhelong Wang, Long Liu, Yi An, Hongyu Zhao, Xin Miao, Ruichen Liu, and Giancarlo Fortino. 2022. Multi-sensor information fusion based on machine learning for real applications in human activity recognition: State-of-the-art and research challenges. *Information Fusion* (2022).
[63] Mohd Sabra, Anindya Maiti, and Murtuza Jadliwala. 2018. Keystroke inference using ambient light sensor on wrist-wearables: a feasibility study. In *ACM WearSys*.
[64] Allen Sarkisyan, Ryan Debbiny, and Ani Nahapetian. 2015. WristSnoop: Smartphone PINs prediction using smartwatch motion sensors. In *IEEE WIFS*.
[65] Leonid Sigal, Alexandru O Balan, and Michael J Black. 2010. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International journal of computer vision* (2010).
[66] Shirish Singh, Devu Manikantan Shila, and Gail Kaiser. 2019. Side channel attack on smartphone sensors to infer gender of the user. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*.
[67] Chen Song, Feng Lin, Zhongjie Ba, Kui Ren, Chi Zhou, and Wenyao Xu. 2016. My smartphone knows what you print: Exploring smartphone-based side-channel attacks against 3d printers. In *ACM CCS*.
[68] Sophie Stephenson, Bijeeta Pal, Stephen Fan, Earlence Fernandes, Yuhang Zhao, and Rahul Chatterjee. 2022. SoK: Authentication in Augmented and Virtual Reality. In *IEEE S&P*.
[69] Tan-Hsu Tan, Jie-Ying Wu, Shing-Hong Liu, and Munkhjargal Gochoo. 2022. Human activity recognition using an ensemble learning algorithm with smartphone sensor data. *Electronics* (2022).
[70] Yin Tang, Lei Zhang, Fuhong Min, and Jun He. 2022. Multiscale deep feature learning for human activity recognition using wearable sensors. *IEEE Transactions on Industrial Electronics* (2022).
[71] Dipanwita Thakur and Suparna Biswas. 2022. An integration of feature extraction and guided regularized random forest feature selection for smartphone based human activity recognition. *Journal of Network and Computer Applications* (2022).
[72] Rahmadi Trimananda, Hieu Le, Hao Cui, Janice Tran Ho, Anastasia Shuba, and Athina Markopoulou. 2022. OVRseen: Auditing Network Traffic and Privacy Policies in Oculus VR. In *USENIX Security Symposium*.
[73] Waseem Ullah, Amin Ullah, Tanveer Hussain, Khan Muhammad, Ali Asghar Heidari, Javier Del Ser, Sung Wook Baik, and Victor Hugo C De Albuquerque. 2022. Artificial Intelligence of Things-assisted two-stream neural network for anomaly detection in surveillance Big Video Data. *Future Generation Computer Systems* (2022).
[74] Roberta Vrskova, Robert Hudec, Patrik Kamencay, and Peter Sykora. 2022. Human activity classification using the 3DCNN architecture. *Applied Sciences* (2022).
[75] Chen Wang, Xiaonan Guo, Yan Wang, Yingying Chen, and Bo Liu. 2016. Friend or foe? Your wearable devices reveal your personal pin. In *ACM ASIACCS*.
[76] He Wang, Ted Tsung-Te Lai, and Romit Roy Choudhury. 2015. Mole: Motion leaks through smartwatch sensors. In *ACM MOBICOM*.
[77] Lin Wang, Junbao Zhang, Yue Li, and Haoyu Wang. 2023. AudioWrite: A Handwriting Recognition System Using Acoustic Signals. In *IEEE ICPADS*.
[78] Mika Westerlund. 2019. The emergence of deepfake technology: A review. *Technology Innovation Management Review* (2019).
[79] Raveen Wijewickrama, Anindya Maiti, and Murtuza Jadliwala. 2019. deWristified: handwriting inference using wrist-based motion sensors revisited. In *ACM WiSec*.
[80] Raveen Wijewickrama, Anindya Maiti, and Murtuza Jadliwala. 2021. Write to know: on the feasibility of wrist motion based user-authentication from handwriting. In *ACM WiSec*.
[81] John Wojewidka. 2020. The deepfake threat to face biometrics. *Biometric Technology Today* (2020).
[82] Yi Wu, Cong Shi, Tianfang Zhang, Payton Walker, Jian Liu, Nitesh Saxena, and Yingying Chen. 2023. Privacy Leakage via Unrestricted Motion-Position Sensors in the Age of Virtual Reality: A Study of Snooping Typed Input on Virtual Keyboards. In *IEEE S&P*.
[83] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. Modeling tabular data using conditional GAN. *Advances in Neural Information Processing Systems* (2019).
[84] Weitao Xu, Girish Revadigar, Chengwen Luo, Neil Bergmann, and Wen Hu. 2016. Walkie-talkie: Motion-assisted automatic key generation for secure on-body device communication. In *ACM/IEEE IPSN*.
[85] Zhi Xu, Kun Bai, and Sencun Zhu. 2012. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *ACM WiSec*.
[86] Lijun Yu, Yijun Qian, Wenhe Liu, and Alexander G Hauptmann. 2022. Argus++: Robust real-time activity detection for unconstrained video streams with overlapping cube proposals. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*.
[87] Tuo Yu and Klara Nahrstedt. 2019. Shoeshacker: Indoor corridor map and user location leakage through force sensors in smart shoes. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* (2019).
[88] Jerrold H Zar. 1972. Significance testing of the Spearman rank correlation coefficient. *J. Amer. Statist. Assoc.* (1972).
[89] Shijia Zhang, Yilin Liu, and Mahanth Gowda. 2023. I Spy You: Eavesdropping Continuous Speech on Smartphones via Motion Sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* (2023).