TapID: Rapid Touch Interaction in Virtual Reality using Wearable Sensing

Manuel Meier* Paul Streli* Andreas Fender* Christian Holz*

Department of Computer Science
ETH Zürich, Switzerland

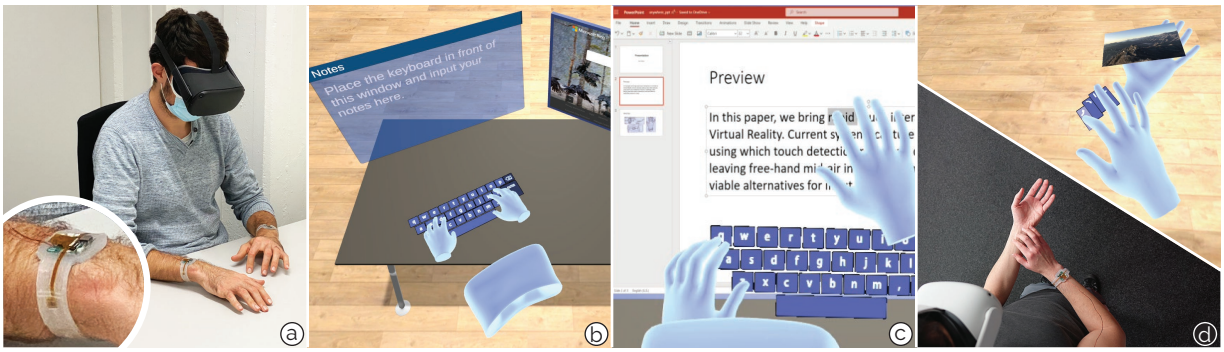

Figure 1: TapID is a wrist-worn device that detects taps on surfaces and identifies the tapping finger, which, combined with tracked hand poses, triggers input in VR. (a) The user is wearing two TapID bands for (b) touch interaction with surface widgets in VR, e.g., for text input, web browsing, or (c) document authoring using familiar front-end apps. (d) Widgets can also be registered to the body itself, using TapID to detect on-body taps and identify the tapping finger, here to rotate an image held in hand.

Abstract

Current Virtual Reality systems typically use cameras to capture user input from controllers or free-hand mid-air interaction. In this paper, we argue that this is a key impediment to productivity scenarios in VR, which require continued interaction over *prolonged* periods of time—a requirement that controller or free-hand input in mid-air does not satisfy. To address this challenge, we bring rapid touch interaction on surfaces to Virtual Reality—the input modality that users have grown used to on phones and tablets for continued use. We present TapID, a wrist-based inertial sensing system that complements headset-tracked hand poses to trigger input in VR. TapID embeds a pair of inertial sensors in a flexible strap, one at either side of the wrist; from the combination of registered signals, TapID reliably detects surface touch events and, more importantly, *identifies* the finger used for touch. We evaluated TapID in a series of user studies on event-detection accuracy ($F_1 = 0.997$) and hand-agnostic finger-identification accuracy (within-user: $F_1 = 0.93$; across users: $F_1 = 0.91$ after 10 refinement taps and $F_1 = 0.87$ without refinement) in a seated table scenario. We conclude with a series of applications that complement hand tracking with touch input and that are uniquely enabled by TapID, including UI control, rapid keyboard typing and piano playing, as well as surface gestures.

Index Terms: Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Virtual reality; Human-centered computing—Human computer interaction (HCI)—Interaction techniques—Gestural input

1 Introduction

The latest Mixed Reality systems incorporate hand pose recognition for input detection, for Augmented Reality (e.g., Hololens 2) and Virtual Reality (e.g., Quest 2) alike. While previous device generations had largely relied on hand-held controllers for input, the transition to controller-free, hand pose and gesture-operated input is apparent, both in the research communities (e.g., [47]) as well as the commercial domain (e.g., [11, 12, 33]).

In Augmented Reality, hand tracking is well-suited to accompany the work on physical objects, for example in maintenance and repair [21] or manufacturing and assembly [42]. Working on physical objects allows the user to hold onto something, thus preventing fatigue during use [22]. In contrast, for the interaction with objects in Virtual Reality, where content is intangible outside passive haptics systems (e.g., [9, 24, 30]), free-hand interaction can become a burden once it exceeds quick interactions [25].

In this paper, we propose moving all *direct* interaction in VR to passive surfaces. Compared to mid-air interaction, touch interaction on surfaces provides users with an opportunity to rest their arms between interactions while simultaneously offering physical support during prolonged interactions. At the same time, the surface provides haptic feedback alongside a frame of reference for proprioception, affording quick and precise interaction.

The core challenge of touch interaction in VR systems is the precise detection of touch events; because hands are tracked from the head-mounted systems themselves, this single vantage point can lead to inaccuracy in depth sensing and thus make touch/no-touch discrimination and contact locating [2] challenging. A variety of approaches have been proposed to detect touch on surfaces using depth cameras, for example for stationary [46] and headset-integrated systems [47]. To combat noise in-depth measurements, they integrate filters for reliable touch detection and rejection. Both, adding filters as well as the framerate of the used camera inherently limit the speed of interaction that systems can reliably detect.

We address the challenge of reliable and quick touch detection with our band TapID, which users wear around their wrist. Through built-in inertial sensors, TapID does not just detect touch events on passive surfaces but additionally identifies the finger used for touch. TapID thus complements the headset's optical tracking of hands and fingers, with which we combine TapID's detected touches to trigger input events in VR. Our approach, therefore, allows applications and interaction modalities known from tablets and phones to seamlessly and reliably transfer to VR scenarios.

*e-mail addresses: firstname.lastname@inf.ethz.ch

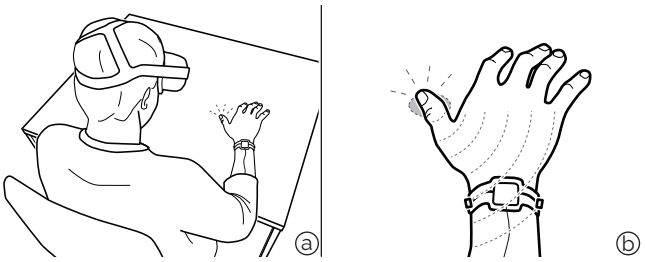

Figure 2: TapID's sensing principle. The user interacts through touches on the table. (a) TapID detects the occurrence of a tap and identifies the tapping finger, while the VR headset tracks fingertip positions. (b) A tap is mechanically propagating through the hand and is registered by TapID's dual-IMU sensor, here identifying the thumb.

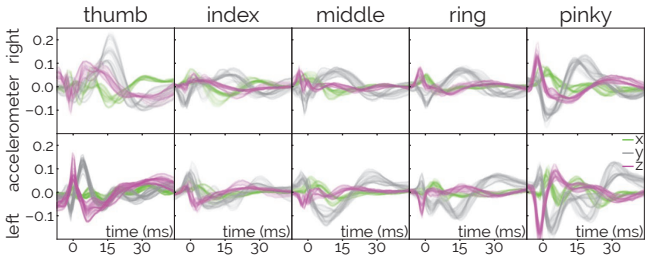

Figure 3: Overlaid inertial measurements from the IMUs on the left and right side of the wrist, respectively, for 30 sample taps caused by each of a user's 5 fingers of the right hand.

1.1 Rapid touch interaction on passive surfaces in VR

Figure 1 shows an overview of our approach. Here, a user is interacting through freehand gestures in VR, while TapID supplies the tracking signals for finger touches on the table in front of them. The user thereby wears one TapID band on either wrist, which streams inertial events to a PC for tap detection and finger identification.

Simultaneously, the VR headset tracks the wearer's hands using built-in vendor algorithms that process the headset's camera feeds. Our system then obtains the 3D-tracked hand poses from the headset for both hands when they are in the view frustrum.

When TapID detects tap events through the inertial sensors, our system forwards the input event to the VR application. After verifying whether the user's hands are within proximity of a passive surface such as a table, our VR app then trigger an input event at the corresponding finger's location. If the user's hands are not close to a surface, our system simply rejects the event detected by TapID as inadvertent input and ignores it.

Figure 1b shows an application that particularly benefits from our approach: a keyboard that affords rapid and reliable text entry using all ten fingers. Upon typing on the keyboard, our system forwards all key input to the operating system for text processing and retrieves word suggestions that the user may select—similar to how they would interact on a regular touch tablet. Our system integrates a browser interface and forwards all input to the renderer, so that any web application such as search engines, word processors, or presentation apps can become usable through touch in VR using TapID. As shown in Figure 1c, Microsoft PowerPoint is running in VR and supports all the functionality users would expect from touch interfaces, such as tapping to place the cursor, double-tapping to select a word, tool buttons, etc. Finally, Figure 1d shows an example application that uses the user's own body for input affordances, for example when passive surfaces around the user are absent. Here, menus and other interactive elements are registered with the user's arm, tracked by the headset, and a tap on the body delivers input to the UI elements, as it is picked up and processed by TapID.

Figure 2 illustrates our sensing principle. The elastic silicone-based strap embeds our SoC platform that connects to two accelerometers, one on either side of the wrist. The accelerometers mechanically couple to the wearer's skin and wrist bones through the strap and, thus, pick up body-coupled vibration signals. During each tap event, the inertial sensors register slightly different vibration signals due to the propagation path through the user's hand depending on the finger they originate from. These differences are subtle yet systematic as shown in Figure 3, and they are the source of our signal, which TapID analyzes using a machine-learning pipeline to detect taps and identify the tapping finger.

In our user study, we evaluated TapID's performance of tap event detection and finger identification with 18 participants. TapID reliably detected tap events ($F_{1,\text{tap}} = .996$) and reliably identified the tapping finger (cross-session $F_{1,\text{finger}} = .93$, cross-person with refinement $F_{1,\text{finger}} = .91$, cross-person without refinement $F_{1,\text{finger}} = .87$).

We compared our results to the performance of our method running on a rigid and *board-mounted* dual-accelerometer configuration (e.g., found in FITBIT bands, such as Charge [43]), where it achieved comparable accuracies for tap detection $F_{1,\text{tap}} = .996$ and worse but still useful finger identification (cross-session $F_{1,\text{finger}} = .92$, cross-person with refinement $F_{1,\text{finger}} = .89$, without refinement $F_{1,\text{finger}} = .80$). We also evaluated our prototype in a traditional SMARTWATCH configuration with only a single board-mounted accelerometer. This setup achieved comparable results to the FITBIT for both tap detection ($F_{1,\text{tap}} = .996$) and finger identification (cross-session $F_{1,\text{finger}} = .91$, cross-person with refinement $F_{1,\text{finger}} = .89$, without refinement $F_{1,\text{finger}} = .81$). These promising results, particularly in the configuration of existing wearable devices, validate our method for the practical use in VR scenarios.

1.2 Contributions

We make three main contributions with our work. 1) We introduce a wrist-based sensing principle to detect surface touches and to identify the finger that has caused a touch through a dual-accelerometer configuration integrated into a watch strap. 2) We demonstrate our prototype implementation TapID that processes and classifies input events in real-time using a machine-learning pipeline. The detected and classified events are then forwarded to our VR system that combines them with the hand posed tracked in 3D by the headset to trigger input in VR. 3) In a technical evaluation with 18 participants, we assess TapID's performance in detecting tap events and correctly identifying fingers. We thereby compare TapID in three configurations: the proposed dual-IMU implementation on the wrist strap, a dual-IMU sensor rigidly mounted on the board (FITBIT), and a single on-board accelerometer (SMARTWATCH), investigating accuracies cross-session and cross-person. Notably, we demonstrate that our method is *neither person-specific nor hand-specific*.

Collectively, these contributions highlight the promise of touch interaction in VR, using the input modality that is familiar from tablet and phone devices, but extending and integrating it into the spatial user interfaces common in VR. We believe that our method has the potential to bring prolonged interaction to VR in the context of creativity and productivity scenarios, where reliable, precise, and non-fatiguing input takes precedence over the shorter and often whole-body engaging interactions that have already proven successful in the context of games and entertainment.

2 RELATED WORK

TapID is related to efforts on input event detection, particularly using wearable sensors, finger identification, and VR interaction.

2.1 Wearable sensors and body-coupled events

Several projects have investigated body-coupled mechanical events for interaction. Approaches mainly differ in the origin of such events,

varying between events that result from hand-held tools, repetitive motions during an activity, contact with passive affordances, and mechanical events caused by tapping one's own body.

Inertial measurement units (IMUs) are frequently used to detect human motion and activity due to their integration into phones and wearable devices [44]. For example, Viband operates the IMU in a watch at a high sample rate to detect handheld tools and manual activities [32]. With the emergence of neural network-based processing pipelines, various network architectures have been presented for such recognition [27, 36, 37, 50], which has started to supersede the previously hand-crafted features (e.g., [13, 26].

To sense explicit interaction with passive affordances, IMUs and microphones have been used to detect touch events (e.g., [6, 18]), which can be spatially localized on the surface using multiple sensors (e.g., [16, 38]). Researchers have also examined body locations for their suitability to mount sensors, such such as fingers (e.g., [18,51]), knuckles (e.g., [1, 35]), or the wrist (e.g., [6, 16]).

Other projects have instead researched tap detection on the user's own body. Such projects often involve sensors arrays to detect bioacoustic waves and are mounted on the forearm (e.g., for forearm and hand taps [20]) or the wrist (e.g., for taps on finger knuckles and other body parts [8, 49]). Beyond tap events, AudioTouch uses two piezoelectric elements on the back of the hand to resolve touch force and classifies 12 hand poses during touch [31]. Similar to TapID, Actitouch complements VR interaction to detect touch input on the body [52], using a wearable with skin contact to modulate an RF signal onto the body that a receiver picks up to detect on-body touch, while the headset supplies locations through hand tracking.

2.2 Finger identification during touch

Several efforts have shown the relevance of finger identification during interaction. On optical tabletops, such efforts used fiducial markers [34] for finger-specific UI control or fingerprint scanning [23] for authentication during interaction. Gupta and Balakrishnan instead mounted optical sensors to the fingertips to distinguish two fingers, using the new capability to shrink touchscreen keyboards in half and overloading each key with two letters for finger-specific input.

Body-worn IMUs are capable of finger identification when mounted close to the source, i.e., the individual fingers. Whichfingers detected tapping fingers from piezoelectric transducers, in one embodiment placed on the fingertips and in another mounted on the knuckles [35]. Similar in design, TapStrap is a consumer product with 5 loosely coupled rings that embed IMUs [1]. Using them, TapStrap implements a one-handed chorded keyboard through tap detection on passive surfaces. Booth and Goldsmith's wrist-worn 8-sensor array along the bottom of the wrist detected gesture input [7], although finger detection accuracies were below .55 across users.

Many scenarios benefit from unencumbered hands and fingers, which has brought about alternative sensing approaches. For example, Becker et al. used a Myo band worn by the elbow and detected individual fingers from electromyography signals [4], reaching accuracies up to .75 (within-user), which slipped below .50 across users. Also using EMG, Benko et al. classified two fingers with .91 accuracy and demonstrated use-cases on touch tables [5].

Having studied the related work in these areas, we conclude that previous approaches show the promise of finger-specific interaction. However, they typically trade off convenience (i.e., positioning sensors close to the fingers or requiring the user to wear a glove) with accuracies that are suitable for reliably distinguishing input from five fingers. We believe that TapID is an encouraging approach in this regard, mounting sensors by the user's wrist inside a convenient and socially acceptable form factor. TapID reliably reaches high cross-user detection accuracies after minimal refinements, which adequately sets up our approach for interactive use.

2.3 Touch interaction in VR

Surface touch interaction has found interest in the Mixed Reality communities. For the purpose of text entry in VR, a commercial high-fps Optitrack system proved as a useful input platform with 23 motion-capture markers affixed to each hand [29].

Without a stationary multi-camera setup in an MR scenario, the temptation has been to use the cameras built into headsets to simultaneously estimate hand poses and detect touch events. For example, MRTouch combines the depth and infrared cameras in a Microsoft Hololens for real-time surface touch detection [47] and discussed the main challenges in their evaluation: the high rate of missed touches (3.5%) and spurious extra touches (19%).

Detecting touch using cameras has been a long-standing challenge. On his Digital Desk, Wellner explored this ambition in the early 90s [45], but concluded that reliable contact sensing required surface instrumentation. Agarwal et al. detected touch locations and moments of contact with overhead stereo cameras [2], discussing how the noise from stereo depth causes erroneous proximity readings. Wilson later introduced an approach that built on a single depth camera, which provided more reliable detection and also scaled to mobile scenarios and on-body input (e.g.,Imaginary Phone [19]).

These camera-based approaches share the benefit of recognizing accurate touch *locations*, but all the struggle with noisy depth estimates for reliable *event detection* and their constraints on input speed that is limited by camera framerates and applied filters. This challenge is what we seek to complement with TapID, using wrist-worn inertial sensors that deliver the missing piece of information on rapid touch interaction: precise events and finger identification.

TapID also derives inspiration from Substitutional Reality, i.e., adopting passive affordances in the environment for VR applications [40]. Such affordances have been used to provide otherwise missing haptic feedback in response to input events (e.g., Sparse Haptic Proxy [9]). The settings that we envision TapID to be used in follow our previous work on situating virtual reality interactions in a different physical reality (e.g., VRoamer [10] and Dreamwalker [48]).

3 TapID Wearable Device and Electronics

TapID precisely detects taps on a surface and identifies the tapping finger, made possible by the integration of our custom wrist-band device. Users wear two such devices (Figure 1a), which then operate independently per hand. Figure 4 shows our electronics platform, which centers around a System-on-a-Chip (DA14695, Dialog Semi) that samples all inertial sensors at a frequency of 1344 Hz and streams the IMU data to a PC through a serial connection. TapID features two low-power accelerometers (LIS2DH, STMicroelectronics) embedded into a wrist strap through a flexible PCB as well as two on-board accelerometers for testing and comparing different sensor configurations. We cast TapID's strap using Shore-32 silicone and embedded all electronics during curing.

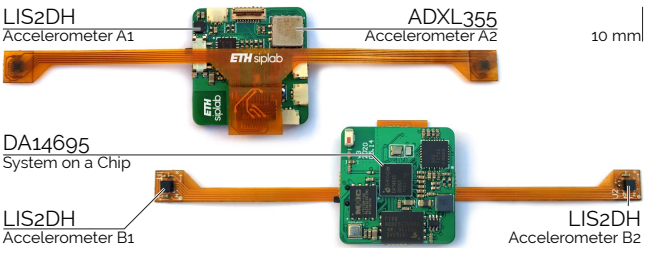

Figure 4: TapID electronics platform. The main PCB features a microcontroller that reads out the two low-power IMUs that are mounted on the ends of a flexible PCB. For comparison in our evaluation, two additional IMUs are mounted on the main PCB.

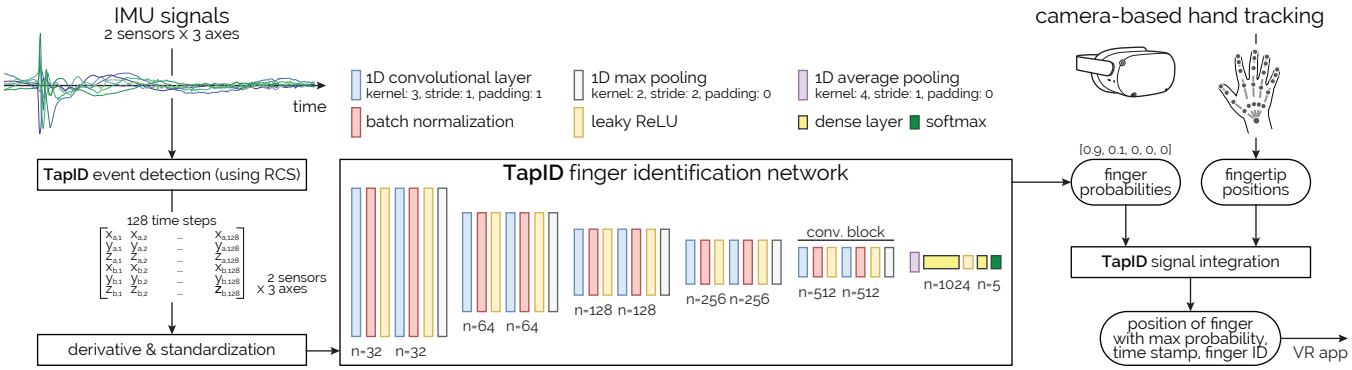

Figure 5: TapID's processing pipeline. Our primary input is the continuous stream from two 3D accelerometers, from which TapID first detects the occurrence of a tap using a rate-of-change score. Upon detection, we extract a window of 128 samples, centered on the tap event, and input it into our neural network classifier to identify the tapping finger. Our architecture comprises five blocks of convolutional layers and two final linear layers that feed into a softmax activation function, which renders a probability distribution over the fingers. We then combine the classification with our secondary input: continuous hand tracking. Having determined which finger has caused the event, we retrieve its fingertip location and forward the tap event to the VR application.

4 Tap Event Processing Pipeline

Our system detects and represents all tap events through a timestamp, the identity of the finger that has caused the tap, and the 2D location on the surface in world coordinates. The system passes this record to the VR app whenever touch events occur. In this section, we describe the three interdependent steps for determining each entry as illustrated in Figure 5. This includes our signal processing for tap detection, our machine learning-based identification pipeline, and our combination with the hand tracking from the VR headset.

4.1 Tap event detection & timestamp calculation

To distinguish tap events from other arm and hand movements, we select events by the sharp spikes that they generate in the accelerometer signal. Rather than performing naïve magnitude thresholding, TapID accumulates the rate of change in the raw signals (*RS*) of all accelerometer axes to calculate a rate-of-change score (*RCS*). To this, we add an exponential decrease to focus on fast changes on the signal only. TapID detects tap events using a threshold on *RCS* peak prominence dependent on the last local minimum.

$$RCS_t = \frac{RCS_{t-1}}{1.6} + \sum_{\text{sensors}} \sum_{\text{axes}} |RS_t - RS_{t-1}| \qquad (1)$$

We derive the exact timestamp of a tap event as the highest *RCS* spike in a window of 20 samples (15 ms) after a threshold has been exceeded. After each event, our processing incurs a back-off period of 200 ms before detecting fresh events. This prevents a single tap from triggering multiple events while still allowing up to 600 input events per minute when using both hands.

4.2 Tap event finger identification

Upon detecting a the occurrence of tap event including its refined timestamp, TapIDforwards a symmetric window of raw accelerometer samples around the event's timestamp to our neural network classifier to identify fingers. Our classifier estimates the likelihood of each finger to have caused the recorded signal. Using supervised learning, we train the network by adjusting its weights via back-propagation to minimize the cross-entropy loss on a training dataset that has a single finger label assigned to each recorded tap event. Note that we merely train on five such classes, one for each finger, creating a classifier that is independent of individual hands.

We observed the best performance on the given task with our classifier implemented as a multi-layer feed-forward 1D-convolutional neural network following a VGG-style architecture [41]. Figure 5 shows our convolutional architecture, which is a suitable choice given the fixed-length of the input window and the strength of convolutional layers to capture local dependencies in a signal. To increase the receptive field, we make use of a deeper network with smaller kernels. Intuitively, this enables the network to extract more powerful features due to the additional non-linearities between layers compared to a shallow network with larger kernels. Specifically, the network consists of five blocks, each made up of two convolutional layers with a kernel size of 3 that are connected to a max pooling layer to reduce the input along the time dimension by a factor of 2. The convolutional blocks are followed by an adaptive average pooling layer and two linear layers. The output of the final linear layer feeds into a softmax activation function that estimates a probability distribution over the five fingers of the hand. We represent the accelerometer data in the time domain as an input tensor of dimensions [*stacked axes of sensors* $\times$ *window size*], where the convolution is applied over the time axis and the per-frame stacked sensor data are treated as input channels.

We apply several data pre-processing steps to boost TapID's performance. Taking advantage of the symmetry between the left and the right hand, we mirror the data of the left hand to receive a comparable input signal from both hands. We swap the data of the symmetrically arranged accelerometers and invert the x or y axis depending on the sensor's orientation. This reduces the variability of the combined dataset and aids the learning process of our classifier, which can now be applied to the signals from either wristband.

Next, we estimate the jerk experienced by each sensor by applying the forward difference operator along each time axis. This operation can be thought of as finding the discrete derivative of the signal, which amplifies high-frequency components in the Fourier domain that we argue contains the discriminative finger information we seek. This step also reduces the lower-frequency artifacts caused by arm and hand motions. Finally, we standardize the resulting segments along the time dimension of each channel across the dataset.

The complete classification network comprises more than 2.1 million trainable parameters. We implemented the model in *Pytorch* and trained it over 30 epochs with a batch size of 16, using the Adam optimizer [28] with a learning rate of 10^{-4}, $\varepsilon = 10^{-8}$ and $\beta = (0.9, 0.999)$. Furthermore, we experimented with different segment sizes around the peak event and achieved the best performance with a window length of 128 frames ($\sim$95 ms). We optimized these hyperparameters over a dataset spanning multiple users and sessions.

4.3 Combining hand tracking with TapID events

When using TapID in a widget-based VR user interface (e.g., those using buttons, sliders, checkboxes, etc.), we need to retrieve the location of a tap following TapID's detection of its occurrence. To achieve this, we complement TapID's classification output with commodity hand tracking that is integrated into contemporary headsets. Having determined which finger has caused a tap, we retrieve the fingertip position from the set of tracked hands and assign it in TapID's *tap event*. If the tip of the detected finger is not within proximity of a surface in VR, our system simply rejects detected events as inadvertent input. In our experiments, we found 30 mm to be a viable maximum surface distance for reliable detection and rejection. Valid surfaces that trigger an input event in VR include the user's own body, such that our system can detect on-body touch input when UI elements are anchored to the user's arm (Figure 1d). In such cases, our system compares fingertip positions with the location of arm surfaces, which we approximate from the 3D-tracked joint with a constant radius (40 mm) plus the delta region mentioned above.

Our VR system is implemented using Unity's 3D game engine for rendering and Velt [15] for the overall data flow. An Oculus Quest headset renders our VR applications and its SDK comes with built-in inside-out tracking for headset positions and hand poses.

While we have found the tracking data from the SDK to have sufficient accuracy for determining the user's fingertip positions in our experiments, it does not afford detecting touches on passive surfaces with any practical reliability or on-body touch input with any reliability using inside-out tracking alone. More severely, rapid finger taps manifest themselves as shaking motions in the tracked hands, though not in position updates that allow for touch detection. Tracking such events becomes even less suitable when fingers are partially occluded from the cameras' perspective, for example when finger is occluded by other fingers during a down motion of the hand. We have also found that the fingertip positions slightly lag behind the true finger positions, which is tolerable for retrieving positions on the surface, but not for detecting touch input or retrieving the exact timestamp of a touch event during interaction.

For those reasons, we merely extract fingertip positions from the hand tracking, but rely on TapID for detecting tap events and revealing the tapping finger. Our system then triggers the event at the 2D position obtained from projecting the fingertip position onto the respective surface where the UI is situated.

4.4 Pipeline latency

Our combination of sensing and VR-enabled hand tracking introduces several sources of latency. The continuous hand tracking itself entails some latency, which depends on the VR device and the implemented hand tracking approach.

With regard to the latency incurred by our own processing, the *total* latency from the moment of the physical tap to displaying its effect in Unity is approximately 130 ms. We estimated this value by manually counting frames in high-frequency video recordings. This duration encompasses the latency caused by all hardware operation, communication, processing, and output.

We empirically determined the latency of TapID's sensing, onboard processing, and communication pipeline, which averaged out to ∼10 ms. The machine learning-based classification including our buffered data acquisition of each tap event adds 50–60 ms. Thus, the latency for detecting a tap and identifying the finger comprises:

1. Data in sensor FIFO: 3 ms (max: 6 ms)
2. SoC processing and serial transmission to host computer: 3 ms
3. Tap event refinement (waiting for highest amplitude): 15 ms
4. Buffer until the data window is complete: 47 ms (64 samples)
5. Neural network inference time 2 ms (*GeForce GTX 1050 Ti*)

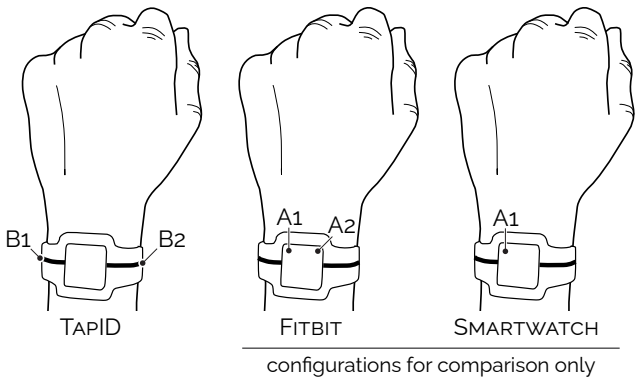

Figure 6: By default, TapID derives its signal from the IMUs B_1, B_2, which are embedded inside the flexible silicone strap at either side of the wrist (TapID configuration). For comparison in our evaluation, we included the two IMUs A_1, A_2 on the main PCB akin to Fitbit's Charge [43]. In the Smartwatch configuration, only A_1 is active. Lastly, we used All 4 IMUs as a baseline for comparison (not shown).

Note that Steps 3 and 4 overlap by around 10 ms on average, as the refined peak usually lies at the beginning of the refinement window. In addition, the latency of transmitting data through serial depends on buffer settings on the host computer.

Taken together, our combination of low-latency tap input with inside-out fingertip tracking is what uniquely enables users to reliably interact with VR applications through touch, using the rapid input they know from regular touch devices. We demonstrate several use-cases that leverage our low-latency approach in Section 6.

5 Technical Evaluation

The goal of this evaluation was to verify our sensing principle and signal processing approach of detecting finger tap events and identifying the finger causing it. We also aimed to determine to what extent wrist-mounted accelerometers in different locations and combinations contribute to the accuracy of our method.

For this purpose, we compared the F_1 scores of classifying inertial data in the three configurations shown in Figure 6: TapID (two accelerometers embedded in the elastic silicone strap), Fitbit (two accelerometers rigidly mounted on the main PCB), Smartwatch (a single low-noise board-mounted accelerometer). We also report the F_1 scores for a baseline where All 4 sensors were simultaneously used. The second purpose of the data capture was to establish a corpus for training the cross-person tap classifier that we used for online operation in our demo apps.

We conducted the evaluation in two parts; In Part 1, we investigated the accuracy of tap event detection. In Part 2, we assessed the accuracies achieved by our finger identification network. Both parts of the evaluation shared the same task and procedure, but were conducted with different participants.

5.1 Task

Participants comfortably sat in an office chair in a table setting, resting their lower arms on the edge of the table, and repeatedly tapped on the tabletop with the finger specified by the experimenter. No other instructions were provided and participants controlled their own tapping frequency, intensity, and tapping locations, finger and hand motions. The experimenter was careful to simply use the word 'tap' to refer to all touch events during the experiment.

5.2 Procedure

The study started with a brief introduction of TapID, which the experimenter related to a regular smartwatch. The experimenter

Sensor configuration	Within-person F_1 scores per finger						Cross-person F_1 scores per finger						Cross-person F_1 scores - 10 tap refinement					
	thumb	index	middle	ring	pinky	macro avg ($\pm\sigma$)	thumb	index	middle	ring	pinky	macro avg ($\pm\sigma$)	thumb	index	middle	ring	pinky	macro avg ($\pm\sigma$)
TapID	0.97	0.92	0.89	0.91	0.94	0.93 ($\pm$0.06)	0.92	0.86	0.83	0.82	0.91	0.87 ($\pm$0.09)	0.95	0.91	0.87	0.89	0.91	0.91 ($\pm$0.07)
Fitbit	0.96	0.92	0.89	0.90	0.93	0.92 ($\pm$0.05)	0.89	0.85	0.70	0.71	0.84	0.80 ($\pm$0.12)	0.96	0.92	0.85	0.83	0.90	0.89 ($\pm$0.07)
Smartwatch	0.97	0.92	0.89	0.87	0.92	0.91 ($\pm$0.05)	0.91	0.86	0.72	0.73	0.83	0.81 ($\pm$0.10)	0.95	0.88	0.86	0.82	0.92	0.89 ($\pm$0.08)
All 4 sensors	0.97	0.94	0.91	0.91	0.94	0.93 ($\pm$0.05)	0.92	0.89	0.77	0.81	0.90	0.86 ($\pm$0.10)	0.96	0.93	0.85	0.91	0.94	0.92 ($\pm$0.08)

Figure 7: F_1 scores for *cross-session* (within-person), *cross-person*, and *cross-person with 10-tap refinement* evaluations. Mean F_1 scores are broken into individual fingers and the mean and standard deviation (σ) of the *Macro F_1 score* across participants.

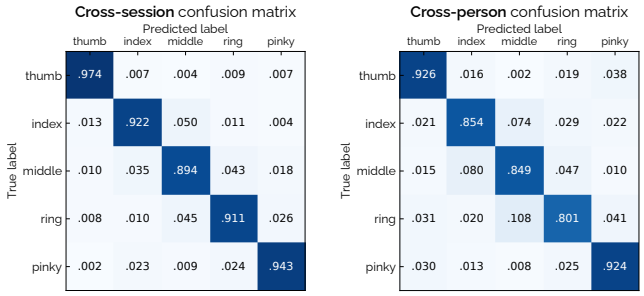

Figure 8: *Left:* Normalized confusion matrix for 3-fold cross-validation for cross-session identification. *Right:* 18-fold cross-validation for cross-person identification.

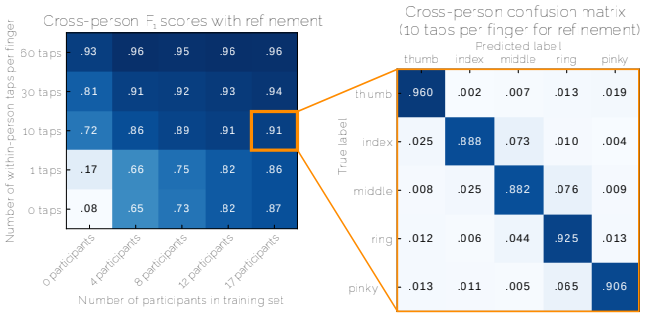

Figure 10: Cross-person refinement accuracy. *Left:* F_1 scores depending on training and refinement set sizes. *Right:* Normalized confusion matrix with 18-fold cross-validation with 10 within-person taps per finger for refinement.

then noted down the participant's age and gender and measured their wrist circumference. Throughout the study, the experimenter instructed participants on which arm to wear the band during each session and which finger to use for tapping each round.

For each round, participants put the TapID device on the specified wrist and repeated 30 taps with each finger, as instructed, before taking off the band again. A block in each session consisted of 30 taps for each finger on either hand, totaling 30×5 fingers $\times$ 2 arms $= 300$ taps per block. Participants performed three such blocks with short breaks in between and produced a total of 300 taps $\times$ 3 blocks $= 900$ taps. Each participant completed the study in under 20 minutes.

5.3 Participants

We recruited 18 participants (14 male, 4 female, ages 19–57, mean = 28.4 years). Participants' average wrist circumference was 168 mm (SD = 10 mm, min = 146 mm, max = 185 mm). Each participant received a small gratuity for their time. Of the 18 participants, 4 (1 female, ages 23–29, mean = 25.3 years) additionally took part in evaluating the performance of our tap detection. Their wrist circumferences ranged from 158 mm to 176 mm.

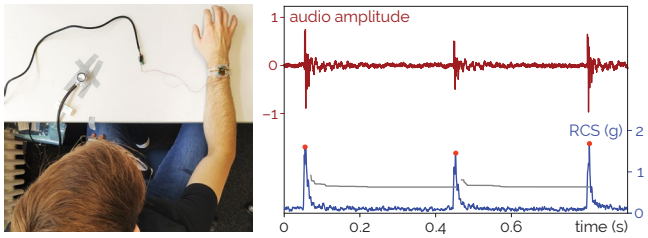

Figure 9: Our evaluation apparatus and procedure. Left: A seated participant, resting his lower arm on the table. The surface-mounted stethoscope served ground-truth for tap events. Right: The resulting audio signal and our Rate-of-Change score to detect events.

5.4 Evaluating tap event detection

To record ground truth annotations for surface taps, we recorded a surface acoustic signal using a stethoscope that was covered and taped to the tabletop (Figure 9). During the evaluation, participants kept quiet to avoid introducing artifacts into the recorded signal.

This sub-evaluation resulted in a total of 3600 tap events. Using our RC score as a measure (Equation 1), our tap detection achieved an accuracy of $F_{1,\text{tap}} = .996$ (precision 99.7%, recall 99.5%) in the TapID configuration. For Fitbit and Smartwatch, the detection achieved $F_{1,\text{tap}} = .996$ as well.

5.5 Evaluating tap event identification

In this sub-evaluation, ground-truth resulted from the experimenter's instructions which finger to use for tapping, which the experimenter logged in our data collection tool for post-hoc label assignment.

5.5.1 Results in TapID configuration

We calculate the average F_1 score for each finger across all participants. Moreover, we report the mean and the standard deviation of the *Macro* F_1 score across all participants, which treats all fingers as being equally important by taking the unweighted mean across class-wise F_1 scores.

Cross-session identification accuracy: We split the recorded data by participant and block. For each participant, we performed 3-fold cross-validation, using two blocks for training and one block for testing while ignoring all data from other participants. Across all participants, the accuracy was, on average, $F_{1,\text{finger}} = .93$ (SD=0.06).

Cross-person identification accuracy: We performed 18-fold cross-validation, testing on each participant's events and training on all other participants. The accuracy was $F_{1,\text{finger}} = .87$ (SD=0.09).

Cross-person identification accuracy with refinement: In addition to pure cross-person testing, we assessed the impact of refinement with individual tap events from a participant's set on the cross-person identification accuracy $F_{1,\text{finger}}$. Simultaneously, we evaluated the impact of the size of the training dataset on the average cross-person identification accuracy.

Fig. 10 shows the mean F_1 scores across participants depending on number of random taps used for refinement (rows) and depending on number of participants' data used for training (columns). For this evaluation, we first trained the classifier for 30 epochs on the recorded taps of (i) 0, (ii) 4, (iii) 8, (iv) 12 and (v) 17 randomly drawn participants—excluding the person in the test set. We then fine-tuned the network for another 30 epochs using only the first (a) 1, (b) 10, (c) 30, and (d) 60 tap events per finger of the tested participant and removed these samples from the test set. We tested the refined classifier on the remaining blocks for that participant.

After fine-tuning, our trained network produced F_1 scores that rose with the number of participants in the training set as well as with the number of refinement taps. With 18-fold cross-validation and training with (iv) all 17 participants' training data, our classifier achieves a mean F_1 score of (a) $F_{1,\text{finger}} = .86$ (SD=0.07), (b) $F_{1,\text{finger}} = .91$ (SD=0.07), (c) $F_{1,\text{finger}} = .94$ (SD=0.05), and (d) $F_{1,\text{finger}} = .96$ (SD=0.05) after corresponding refinement.

5.5.2 Results in other configurations

For our baseline comparisons, Figure 7 lists all the scores achieved with the corresponding sensor configurations alongside a baseline using ALL 4 IMUs. Compared to the FITBIT and SMARTWATCH configurations, TAPID produced consistently higher F_1 scores. In comparison, using ALL 4 IMUs for training and testing added marginal improvements over TAPID and only in some of the cases.

5.6 Discussion

The evaluation confirmed the validity of our sensing principle and our implementation on a wrist-worn device to capture body-coupled events. It also confirmed the TAPID configuration of our prototype, embedding two low-cost, low-power IMUs in the silicone strap.

Regarding the tap detection, the accuracy TapID achieved is high enough for practical purposes, but it also is in FITBIT and SMARTWATCH configuration. This confirms recent results in related efforts (e.g., [18]), though with IMU sensors positioned at the wrist for unencumbered hands instead of placing them on the fingers. One trade-off remains in our tap detection: Decreasing the back-off period allows for faster interactions but increases the likelihood that one (strong) tap produces two events (i.e., one false positive).

As for the performance of our classifier to identify individual fingers, unsurprisingly we saw the best cross-person accuracy when training with the largest amount possible, refined with person-specific training data, reaching F_1 scores of up to .96 this way (Figure 10). What is notable, however, is that a single person-specific calibration procedure holds across recording sessions, setting up novel users with a one-time on-boarding step to reach an accuracy of .96. Even by just fine-tuning with 10 samples, which can easily be recorded in under a minute, TapID identifies the correct finger with an accuracy of $F_{1,\text{finger}} = .91$, notably for any given touch event.

Without refinement, however, cross-person classification achieves an accuracy of $F_{1,\text{finger}} = .87$, which we believe still has room for improvement to rise to practical levels. That said, this F_1 score compares favorably to results achieved in the related work with comparable setups with unencumbered fingers and hands (e.g., [4,7]) for cross-person classification. Interestingly, our method achieves the largest performance increase in TAPID configuration over the FITBIT and SMARTWATCH configurations (Figure 7), while person-specific refinement narrows the gap between configurations.

Taking a look at the performance of individual fingers, our method most reliably identifies thumb and pinky taps, while distinguishing between middle and ring fingers proved to be more challenging. While all finger F_1 scores are $> .89$ for cross-session, the ring finger drops to .82 for cross-person F_1 scores. The right confusion matrix in Fig. 8 illustrates that the model especially struggles to differentiate between the index, middle and ring finger.

Refinement with just 10 taps, however, leads to an improved discrimination between the three inner fingers of the hand and recovers all finger's F_1 scores to values $> .87$ (see Figure 10 and Figure 7).

Inspecting results in detail, physical measures such as a person's wrist circumference may impact the classification accuracy. Performing cross-validation on the 9 participants with a wrist circumference closest to the average of our population, the cross-person classification without refinement rises to $F_{1,\text{finger}} = .92$ for TAPID (compared to $F_{1,\text{finger}} = .82$ when using the other half of our population for validation). Similar trends occurred for all sensor configurations and one reason may be the constant distance between IMUs. Based on these results, we hypothesize that training data from more participants with smaller and larger wrists will decrease this effect but strengthen our cross-person accuracy without refinement.

6 USE CASES

TapID can be used for complementary high-fidelity input in many VR scenarios. In this section, we first describe broader scenarios and envisioned use cases for TapID. We then showcase several apps that we implemented to demonstrate the usefulness of our approach.

6.1 Scenarios

In typical VR scenarios, users interact with non-tangible objects, which lack all physical affordances and, thus, require interaction that is mediated through hand-held controllers or mere hands. However, mid-air input—especially in the case of prolonged interaction—often triggers fatigue, because the lack of physical affordances leaves no options for rest [40]. We argue that, therefore, the main success of VR applications so far has been in gaming that engages the whole-body (e.g., Beat Saber [3]). In contrast, even the most recent productivity applications that attempt to replicate office environments (e.g., Facebook Infinite Spaces [14]) focus on content *consumption* rather than content creation. We believe this is a direct result of the inadequate interaction modality that is mid-air input, often retrofit to an interface that was designed for keyboards, mice, and not least touch. In VR, no devices currently exist to facilitate such interaction other than the handheld controllers, which induce short-term novelty, but have so far not been convincing in enabling long-term productivity, especially when compared to conventional desktop input devices.

We believe that the potential of immersive VR productivity apps remains untapped if the input modality is not effective, precise, and non-fatiguing. Much of productivity work is characterized by sustained interaction, often in the form of fluid and rapid input, often bimanual, and often involving tools to accomplish tasks. Our prototype TapID now offers an alternative interaction concept, which is inspired by interaction in physical offices, but adapted to support potentially infinite virtual spaces of content in VR.

Besides productivity-based scenarios, TapID can also be useful in more playful use-cases that differ from previous (mostly mid-air based) VR experiences. For instance, TapID enables VR games that require surface input with exact timing (e.g., rhythm games).

6.2 Application prototypes

We implemented several applications that showcase our unique combination of rapid tap interaction in VR through TapID in conjunction with spatial interfaces mediated by a VR headset. In these demonstrations, the user is situated in VR and surrounded by content typical in desktop and touch user interfaces, such as photo collections and web browsers. To facilitate interaction with the user interface, we developed a series of *tapping-based virtual input devices* that the user can open and position by tapping the desired location on the surface with specific fingers. For instance, the left pinky finger can summon arrow keys while the the right pinky finger triggers a numeric keypad to appear and the thumb can summon a simple keyboard between both hands.

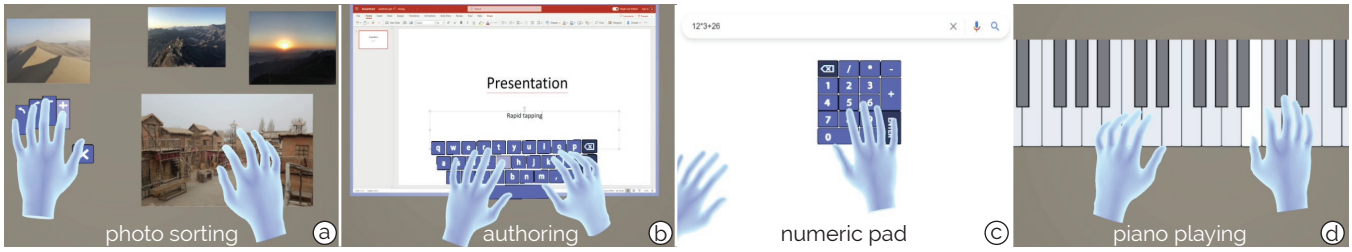

Figure 11: Our demo applications showcase TapID's integration with VR scenes and hand-based interaction using different surface widgets: (a) dragging and resizing photos including a per-finger command input widget below the left hand, (b) editing rich-text documents with keyboard input including text cursor navigation, (c) a numeric keypad for inputting numbers and operations, and (d) virtual piano keys for precise input.

While our applications show hands during interaction, we render fingertips and touches using colored spheres during ego-centric use, showing a line that projects down to the surface and following previous UI recommendations [17]. Screenshots of our applications are shown in Figures 1 and 11.

Photo sorting: Familiar from multi-touch devices, we implemented a *Photo sorting* app as shown in Figure 11a). The app uses TapID's reliable tap detection to trigger drag events for moving and resizing pictures. A command input menu enables step-wise resizing and rotating images. With our approach, we can easily assign these commands to each finger. This prototype also works without a table in an on-body configuration with one wristband worn on one arm and with the user's other arm as tapping surface (see Figure 1d).

Document editing: We designed a simple keyboard input widget to add text to documents (Figure 11b), complementing the use of existing document authoring tools (e.g., Microsoft Word, PowerPoint). Words can be selected through double-tap, e.g., to change their format. Additionally, our arrow key virtual input (summoned with left pinky finger) aids in quickly adjusting the text cursor's desired location. This demonstrates our support of conventional touch capabilities, seamlessly integrating our finger-specific tapping-based virtual input devices with operating system-provided controls and existing applications that were designed for touch.

Number input: Users can open a virtual numeric keypad widget to type in numbers and operations as shown in Figure 11c. This widget conveniently fits below the hand, so that users can quickly input numbers using all five fingers of one hand. This affordance is useful, for example, when summoning and using a calculator as part of the interaction flow in a user interface.

Piano: Our final demonstration is a virtual piano app as shown in Figure 11d. This VR piano optimally leverages TapID for *temporally precise input* and enables users to hit piano keys using any finger. The combination of visual feedback in VR, the passive haptic feedback of the surface, and the low latency of our approach uniquely enables playing this virtual instrument while preserving the exact beat of the user's tapping input.

7 Limitations

Before reaching production-level performance, a couple of limitations remain in our current implementation. Apart from the currently necessary user-specific refinements to achieve reliable accuracies, we have not tuned or evaluated our method on its capability to identify multi-touch events—at this time, only individual fingers are detected, though with simultaneous bimanual input. A second assumption behind our current implementation is the use of a table or the user's own body as a passive input surface We did not experimentally and quantitatively evaluate the detection accuracy of our current implementation on surfaces other than office tables made from coated wood. We also did not explicitly vary or control tapping strength. We have tested and evaluated our system in a stationary and seated setting, without accounting for body motion during use (e.g., in mobile scenarios). With respect to supported input poses, TapID detects and identifies taps with bent fingers. In this case, however, the derived input location may be inaccurate or even unavailable and thus unknown due to our current reliance on the input locations reported from the headset. For our current system to work, the hand producing the input event must be in the tracking area of the VR headset, otherwise input events on the wrist-worn sensor will be dismissed as inadvertent. This entails that, currently, the user needs to look at the content they interact with in VR, including text input on the shown keyboard, which, thus, does not support eyes-free operation. All interaction is also required to be direct, which previous work has found to lead to higher error in 3D scenes [39], but work by Knierim et al. on text input in VR showed to be beneficial, as experienced typists improved their typing performance by seeing their hands and the keyboard in VR [29].

Regarding the suitability of our approach to be implemented on commodity smartwatches, we acknowledge that it may appear unreasonable for a user to wear two smartwatches, which our current approach requires for bimanual operation. However, much like currently used controllers for input, it may seem fathomable that future users put on two small wristbands for all interaction in VR. Finally, our current inference pipeline relies on a moderate GPU and is, at this point, not suitable for embedded processing.

8 Conclusions

We have presented TapID, a wearable sensing device that complements camera-based hand tracking to bring rapid touch interaction to VR systems. Our approach, thus, establish the basis for reliable and prolonged interaction during continuous use, where users may rest their arms and hands on a surface. For each tap on a passive surface, such as a tabletop, TapID reliably detects the event and identifies which finger has been used for the touch. This allows VR systems that track the user's hands in full to add the missing key element: rapid surface contact detection that additionally disambiguates which finger caused the action, such that the VR app can trigger the input event in the application.

In our evaluation with 18 participants, TapID produced convincing accuracies for tap detection and finger identification in a cross-session setting. Cross-validation testing with a classifier that was trained on just cross-person data showed a drop in accuracy, from which TapID's can recover, however, with a small number of a user's tap events for refinement training.

Overall, we believe that the approach we have outlined in this paper is a viable complement for existing VR systems that allows the familiar input modality that is touch input to transfer to spatial VR systems, potentially even during everyday scenarios. A promising insight in this regard is the evaluation of our method in FITBIT configuration, using only TapID's board-mounted IMU sensors as found in existing commodity wearables [43], which produced comparable accuracies when implementing our method.

REFERENCES

[1] 2020. Tap Strap 2: All-In-One Wearable Keyboard, Mouse & Air Gesture Controller. (2020). https://www.tapwithus.com/

[2] Ankur Agarwal, Shahram Izadi, Manmohan Chandraker, and Andrew Blake. 2007. High precision multi-touch sensing on surfaces using overhead cameras. In *Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07)*. IEEE, 197–200.

[3] Beat Games. 2018. Beat Saber. (2018). https://store.steampowered.com/app/620980/Beat_Saber, Accessed: 1/25/2021.

[4] Vincent Becker, Pietro Oldrati, Liliana Barrios, and Gábor Sörös. 2018. Touchsense: classifying finger touches and measuring their force with an electromyography armband. In *Proceedings of the 2018 ACM International Symposium on Wearable Computers*. 1–8.

[5] Hrvoje Benko, T Scott Saponas, Dan Morris, and Desney Tan. 2009. Enhancing input on and above the interactive surface with muscle sensing. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. 93–100.

[6] Riley Booth and Peter Goldsmith. 2017. Detecting finger gestures with a wrist worn piezoelectric sensor array. In *2017 IEEE international conference on systems, man, and cybernetics (SMC)*. IEEE, 3665–3670.

[7] Riley Booth and Peter Goldsmith. 2018. A wrist-worn piezoelectric sensor array for gesture input. *Journal of Medical and Biological Engineering* 38, 2 (2018), 284–295.

[8] Wenqiang Chen, Lin Chen, Yandao Huang, Xinyu Zhang, Lu Wang, Rukhsana Ruby, and Kaishun Wu. 2019. Taprint: Secure text input for commodity smart wristbands. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–16.

[9] Lung-Pan Cheng, Eyal Ofek, Christian Holz, Hrvoje Benko, and Andrew D Wilson. 2017. Sparse haptic proxy: Touch feedback in virtual environments using a general passive prop. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 3718–3728.

[10] Lung-Pan Cheng, Eyal Ofek, Christian Holz, and Andrew D Wilson. 2019. VRoamer: Generating On-The-Fly VR Experiences While Walking inside Large, Unknown Real-World Building Environments. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 359–366.

[11] HTC Corporation. 2016. Vive, Discover Virtual Reality Beyond Imagination. (2016). https://www.vive.com/us/

[12] Microsoft Corporation. 2019. Microsoft HoloLens, Mixed Reality Technology for Business. (2019). https://www.microsoft.com/en-us/hololens

[13] Macarena Espinilla, Javier Medina, Alberto Salguero, Naomi Irvine, Mark Donnelly, Ian Cleland, and Chris Nugent. 2018. Human Activity Recognition from the Acceleration Data of a Wearable Device. Which Features Are More Relevant by Activities?. In *Multidisciplinary Digital Publishing Institute Proceedings*, Vol. 2. 1242.

[14] Facebook. 2020. Infinite Office (trailer). (2020). https://www.youtube.com/watch?v=5_bVkbG1ZCo, Accessed: 1/25/2021.

[15] Andreas Fender and Jörg Müller. 2018. Velt: A Framework for Multi RGB-D Camera Systems. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces (ISS '18)*. Association for Computing Machinery, New York, NY, USA, 73–83. DOI: http://dx.doi.org/10.1145/3279778.3279794

[16] Jun Gong, Aakar Gupta, and Hrvoje Benko. 2020. Acustico: Surface Tap Detection and Localization using Wrist-based Acoustic TDOA Sensing. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 406–419.

[17] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. 2018. Effects of hand representations for typing in virtual reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 151–158.

[18] Yizheng Gu, Chun Yu, Zhipeng Li, Weiqi Li, Shuchang Xu, Xiaoying Wei, and Yuanchun Shi. 2019. Accurate and Low-Latency Sensing of Touch Contact on Any Surface with Finger-Worn IMU Sensor. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 1059–1070.

[19] Sean Gustafson, Christian Holz, and Patrick Baudisch. 2011. Imaginary phone: learning imaginary interfaces by transferring spatial memory from a familiar device. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. 283–292.

[20] Chris Harrison, Desney Tan, and Dan Morris. 2010. Skinput: appropriating the body as an input surface. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 453–462.

[21] Steven Henderson and Steven Feiner. 2010. Exploring the benefits of augmented reality documentation for maintenance and repair. *IEEE transactions on visualization and computer graphics* 17, 10 (2010), 1355–1368.

[22] Juan David Hincapié-Ramos, Xiang Guo, Paymahn Moghadasian, and Pourang Irani. 2014. Consumed endurance: a metric to quantify arm fatigue of mid-air interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1063–1072.

[23] Christian Holz and Patrick Baudisch. 2013. Fiberio: a touchscreen that senses fingerprints. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 41–50.

[24] Brent Edward Insko, M Meehan, M Whitton, and F Brooks. 2001. *Passive haptics significantly enhances virtual environments*. Ph.D. Dissertation. University of North Carolina at Chapel Hill.

[25] Sujin Jang, Wolfgang Stuerzlinger, Satyajit Ambike, and Karthik Ramani. 2017. Modeling cumulative arm fatigue in mid-air interaction based on perceived exertion and kinetics of arm motion. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 3328–3339.

[26] Majid Janidarmian, Atena Roshan Fekr, Katarzyna Radecka, and Zeljko Zilic. 2017. A comprehensive analysis on wearable acceleration sensors in human activity recognition. *Sensors* 17, 3 (2017), 529.

[27] Wenchao Jiang and Zhaozheng Yin. 2015. Human activity recognition using wearable sensors by deep convolutional neural networks. In *Proceedings of the 23rd ACM international conference on Multimedia*. 1307–1310.

[28] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[29] Pascal Knierim, Valentin Schwind, Anna Maria Feit, Florian Nieuwenhuizen, and Niels Henze. 2018. Physical keyboards in virtual reality: Analysis of typing performance and effects of avatar hands. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–9.

[30] Luv Kohli. 2010. Redirected touching: Warping space to remap passive haptics. In *2010 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 129–130.

[31] Yuki Kubo, Yuto Koguchi, Buntarou Shizuki, Shin Takahashi, and Otmar Hilliges. 2019. AudioTouch: Minimally Invasive Sensing of Micro-Gestures via Active Bio-Acoustic Sensing. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services*. 1–13.

[32] Gierad Laput, Robert Xiao, and Chris Harrison. 2016. Viband: High-fidelity bio-acoustic sensing using commodity smartwatch accelerometers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. 321–333.

[33] Facebook Technologies LLC. 2020. Oculus Quest 2: Our Most Advanced All-in-One VR Headset. (2020). https://www.oculus.com/quest-2/

[34] Nicolai Marquardt, Johannes Kiemer, and Saul Greenberg. 2010. What caused that touch? expressive interaction with a surface through fiduciary-tagged gloves. In *ACM International Conference on Interactive Tabletops and Surfaces*. 139–142.

[35] Damien Masson, Alix Goguey, Sylvain Malacria, and Géry Casiez. 2017. Whichfingers: identifying fingers on touch surfaces and keyboards using vibration sensors. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. 41–48.

[36] Abdulmajid Murad and Jae-Young Pyun. 2017. Deep recurrent neural networks for human activity recognition. *Sensors* 17, 11 (2017), 2556.

[37] Francisco Javier Ordóñez and Daniel Roggen. 2016. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors* 16, 1 (2016), 115.

[38] Yilei Shi, Haimo Zhang, Jiashuo Cao, and Suranga Nanayakkara. 2020. VersaTouch: A Versatile Plug-and-Play System that Enables Touch

Interactions on Everyday Passive Surfaces. In *Proceedings of the Augmented Humans International Conference*. 1–12.

[39] Adalberto L Simeone and Hans Gellerseny. 2015. Comparing indirect and direct touch in a stereoscopic interaction task. In *2015 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 105–108.

[40] Adalberto L Simeone, Eduardo Velloso, and Hans Gellersen. 2015. Substitutional reality: Using the physical environment to design virtual reality experiences. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 3307–3316.

[41] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[42] Arthur Tang, Charles Owen, Frank Biocca, and Weimin Mou. 2003. Comparative effectiveness of augmented reality in object assembly. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 73–80.

[43] Leland Teschler. 2016. Teardown: Inside the Fitbit Charge. (2016). `https://www.microcontrollertips.com/inside-fitbit-charge/`

[44] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. 2019. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters* 119 (2019), 3–11.

[45] Pierre Wellner. 1993. Interacting with paper on the DigitalDesk. *Commun. ACM* 36, 7 (1993), 87–96.

[46] Andrew D Wilson. 2010. Using a depth camera as a touch sensor. In *ACM international conference on interactive tabletops and surfaces*. 69–72.

[47] Robert Xiao, Julia Schwarz, Nick Throm, Andrew D Wilson, and Hrvoje Benko. 2018. MRTouch: adding touch input to head-mounted mixed reality. *IEEE transactions on visualization and computer graphics* 24, 4 (2018), 1653–1660.

[48] Jackie Yang, Christian Holz, Eyal Ofek, and Andrew D Wilson. 2019. Dreamwalker: Substituting real-world walking experiences with a virtual reality. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 1093–1107.

[49] Naoya Yoshimura, Takuya Maekawa, Daich Amagata, and Takahiro Hara. 2018. Preliminary Investigation of Fine-Grained Gesture Recognition With Signal Super-Resolution. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 484–487.

[50] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. 2014. Convolutional neural networks for human activity recognition using mobile sensors. In *6th International Conference on Mobile Computing, Applications and Services*. IEEE, 197–205.

[51] Cheng Zhang, Anandghan Waghmare, Pranav Kundra, Yiming Pu, Scott Gilliland, Thomas Ploetz, Thad E Starner, Omer T Inan, and Gregory D Abowd. 2017. Fingersound: Recognizing unistroke thumb gestures using a ring. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (2017), 1–19.

[52] Yang Zhang, Wolf Kienzle, Yanjun Ma, Shiu S Ng, Hrvoje Benko, and Chris Harrison. 2019. ActiTouch: Robust Touch Detection for On-Skin AR/VR Interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 1151–1159.