# WhichFingers: Identifying Fingers on Touch Surfaces and Keyboards using Vibration Sensors

Damien Masson, Alix Goguey, Sylvain Malacria, Géry Casiez

# WhichFingers: Identifying Fingers on Touch Surfaces and Keyboards using Vibration Sensors

**Damien Masson**[1], **Alix Goguey**[2,3], **Sylvain Malacria**[3] **& Géry Casiez**[1]
[1]Université de Lille, France, [2]University of Saskatchewan, Canada, [3]Inria, Lille, France
damien.masson@etudiant.univ-lille1.fr, alix.goguey@usask.ca, sylvain.malacria@inria.fr,
gery.casiez@univ-lille1.fr

## ABSTRACT
HCI researchers lack low-latency and robust systems to support the design and development of interaction techniques using finger identification. We developed a low-cost prototype using piezo-based vibration sensors attached to each finger. By combining the events from an input device with the information from the vibration sensors we demonstrate how to achieve low-latency and robust finger identification. Our prototype was evaluated in a controlled experiment, using two keyboards and a touchpad, showing single-touch recognition rates of 98.2% for the keyboard and 99.7% for the touchpad, and 94.7% for two simultaneous touches. These results were confirmed in an additional laboratory-style experiment with ecologically valid tasks. Last we present new interaction techniques made possible using this technology.

### Author Keywords
finger identification; touch interaction; vibration sensor.

### ACM Classification Keywords
H.5.2 Information interfaces (e.g. HCI): User interfaces

## INTRODUCTION AND RELATED WORK
Finger identification – associating specific fingers and hands with touch contacts on a device – is seeing a rising interest, given the increased input vocabulary it provides [12, 14, 15, 33]. By executing similar physical actions, but with different fingers, the user can input different commands to the system. However, no technology efficiently supports finger identification yet, and as a result, researchers have explored different methods. Gupta *et al.* recently used finger identification to improve typing speed on miniature screens using two fingers [15] and help multitasking on smartphones [14]. Gupta *et al.* acknowledge they explored multiple techniques to identify index and middle fingers, using optical markers, color markers, leap motion, IMU and muscle sensing before ending up using an IR photo-resistor mounted under the index finger, introducing some constraints. In [33], Zheng and Vogel propose to augment keyboard shortcuts by differentiating which finger presses the trigger key. Their prototype requires the use of a green keyboard cover and laptop skin together with a reflector that directs the webcam light path to the keyboard. After complex image processing, they identify any of the ten fingers and two postures (opened and closed hand) within 77 ms but the accuracy of their technique was not reported.

Other prototypes developed in the literature are mainly based on contacts geometry [1, 5, 9, 31, 18, 29], RGB and depth cameras [4, 11, 13, 17, 24, 30, 33], and fingerprints [27, 16].

Geometry-based techniques need no extra hardware but constrain users' movements as they require to put fingers at predefined places [9] or at least three fingers in contact to infer their identity based on geometrical relationships [1, 5, 9, 31, 18, 29]. The reliability of the prototypes is seldom reported. Au and Tai achieved a 98.6% finger identification rate when all 5 fingers of the hand are in contact with a surface [1]. Wagner *et al.* also rely on geometrical features to achieve up to 98.4% accuracy to recognize only nine 3-finger chords [29].

RGB cameras have been used to recognize specific hand postures involving different fingers [17, 21]. Depth cameras can help to segment the hand from the background [24] but as soon as the hand can interact more freely, fingers need to be equipped with color rings to improve the robustness of detection [11, 13, 30]. Leap Motion has been investigated but it requires users to first open the hand such that the algorithm identifies the 5 fingers, and then fold the fingers such that the finger selected for interaction is the furthest away from the others [4]. In addition, it does not work well in front of a screen due to high reflection. Success rates have only been reported in [24], achieving 97.7% for 9 hand poses requiring to hold the hand still for 10 seconds. Overall, camera-based techniques require good lighting conditions, careful camera positioning, pre-calibration and are subject to occlusions.

Sugiura *et al.* used a fingerprint scanner to identify single fingers [27]. In Fiberio [16], the authors designed a touchscreen able to *identify* users by recognizing their fingerprints. They asked participants to lay flat for 400 ms, one after the other, their index, middle and ring fingers. The image captured by the touchscreen could be used to correctly identify the finger, and thus the user, in 98.7% of all cases. Pushing the idea further, the system might be able to identify fingers given that each fingerprint is unique, although it requires the

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

users to hold a finger flat on the surface for at least 400 ms and takes significant time to process. In addition it requires specific expansive hardware that is difficult to adapt to mobile scenarios. Finally, it remains unclear if fingers could be identified well when users interact with their fingertip, only showing a partial fingerprint.

Several other techniques have been explored for finger identification. For instance, gloves augmented with fiduciary markers on a FTIR touchscreen can be used [22], but this cannot be applied to capacitive surfaces. Gupta *et al*. used time-of-flight IR cameras attached to two fingers to measure the distance from the screen and achieved a 99.5% success rate for single touches only [15]. Benko *et al*. used electro-myogram sensors placed on the forearm and achieve a 90% success rate for single finger identification among 5 fingers with a 300 ms latency [2]. Other works attached Gametrak strings to each finger to track their 3D positions [10] or used RFID tags attached to each finger with fake fingernails or gloves on custom touchscreens integrating RFID antennas [19, 28].

Fukumoto *et al*. developed FingeRing [7, 8], a wearable chord keyboard using accelerometers mounted on each finger where users can input chord sequences by tapping with their fingers on any surface. Performance was not measured but when reimplementing the technique, we found that the use of accelerometers capped the recognition accuracy due to false positives when moving the fingers.

When interaction techniques are introduced with a system to identify fingers, most of them bind a command to a single finger [2, 4, 14, 22, 27, 28, 30, 33]. Other applications explore the use of common two-finger gestures (*e.g.* 2-fingers swipes) [2, 13]. However, when it comes to chords with more than two fingers, they use sequential construction of chord (*e.g.* in [9, 11, 13, 14, 20, 29]) which would be captured as successive identification of 1 finger chord.

In summary, none of the previously introduced solutions is concurrently robust, low-latency, cross device and easy to replicate. We argue that finger identification needs to be reliable, fast and able to recognize any finger chord combination. If finger identification is slow, it simply outweighs the benefits of using such technology. If it is unreliable, users are likely to wait for feedback from the system to make sure their fingers are correctly recognized before interacting, slowing down the interaction and reducing the possible benefits of using finger identification. Although we believe that future generations of multi-touch devices will embed a non-invasive, mature and reliable finger identification technology, using capacitive sensing [32, 6] or 3D ultrasound biometric sensors [25], researchers need a robust environment to explore the use of this information to propose useful applications today.

In this paper, we propose WhichFingers, a low-cost device enabling real-time finger identification, using piezo-based vibration sensors. Our device is the first real-time solution which does not require calibration and works not only on any touch surface, but also on keyboards and supports cross-device uses. We are also the first to evaluate a finger-identification prototype in realistic and diverse scenarios. After describing the hardware, we present its evaluation demonstrating its robustness. Last we present new interaction techniques made possible by our device prototype.

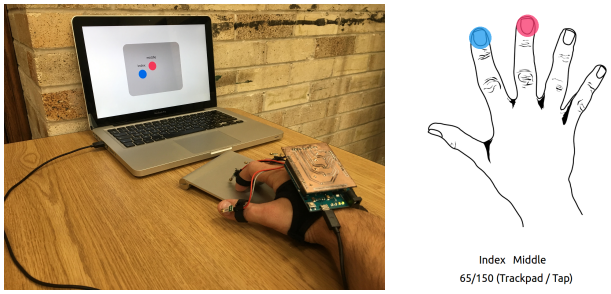


**Figure 1: Our device prototype (left) using vibration sensors attached to the fingers using elastic rings. Example of stimuli (right) used during the controlled experiment.**

## HARDWARE DEVICE

The WhichFingers prototype consists of five Minisense 100 vibration sensors [26] attached to each finger (see Fig. 1-left). The sensors use flexible PVDF piezoelectric polymer film loaded by a mass to offer high sensitivity to detect contact vibrations. They produce a voltage as large as 90V depending on the intensity of the shock or vibration. Our underlying assumption is that the finger that contacts a touch surface or a keyboard key produces a higher response on its sensor compared to the other fingers and thus can be identified.

The five sensors are plugged into a micro-controller through a custom designed shield that connects each sensor to the ground and an analog input of the board. A 1MΩ resistor connected in parallel of each sensor acts as a high pass filter.

We developed a device prototype using an Arduino Leonardo board that reads the voltage of each sensor and sends the raw values to the host computer at 1000 Hz using Raw HID over USB (Fig. 1). In total, the device costs less than $35.

The device is attached on the back of the hand using a fingerless glove. Sensors are glued to elastic rings. To put on the device, users first attach the fingerless glove and then attach each ring to the corresponding finger. The sensors or the elastic rings do not disturb the interaction with a touch surface or keyboard as the finger pulp remains free.

## PROCESSING SOFTWARE

The software that processes the data transmitted by the hardware device has two main components: a *Low-level interaction monitor* and a *Simple signal processing algorithm*.

On desktop, the low-level interaction monitor detects touch and key events on the touchpad and keyboard. It has been developed in C++ with the Qt framework. It monitors raw touchpad inputs using the I/O Kit and Apple's private multitouch API, and key events using the Quartz Event Services API. On mobile Android devices, the low-level interaction monitor detects touch events using native code. Note that WhichFingers can be used with other touch-based devices, as long as a low-level interaction monitor can be implemented.

To identify which finger has been used to perform an operation, we use a simple algorithm that examines which vibration sensor created the highest voltage right before the event occurred. As the software receives the values of the vibration sensor at a frequency of 1000Hz, which is much higher than the frequency of detecting low-level events ($<$ 120Hz), the algorithm first retrieves for an event detected at time $t$ all the vibration values from time $t-32$ *ms* to $t-8$ *ms*, which was empirically defined as the best timeframe for 1000Hz. We estimated that the latency between the time the finger contacts the touchpad or presses a keyboard key to the time the event is reported in our application to be around 30 ms while our wired version reports the information to the host-computer within 1 ms, which helps explain why using this time window works best [3]. Our algorithm then declares the vibration sensor that created the highest voltage over the timeframe as the finger that produced the input event. On touch surfaces, if two (or more) contacts occur in less than 30 *ms*, the two (or more) highest voltages over the overlapping time frames are stored. Fingers are then disambiguated using the $x$ position of the contacts: the leftmost contact is associated with the leftmost finger and the second contact is associated with the remaining finger.

Our processing software handles all touch and key events and receives the raw HID events from WhichFingers.

## CONTROLLED EXPERIMENT

We conducted a *controlled* experiment, targeted at evaluating the robustness of the prototype when the participant was asked to type keys on two different keyboards as well as contacting and performing slides on a touchpad with different combination of fingers.

### Method, procedures and tasks

In this experiment, we asked 20 participants (27.7 mean age, all right-handed, 2 females, computer science university staff and students), equipped with our wired prototype to interact with a keyboard or a touchpad in reaction to a visual stimuli. Vibration sensors were positioned on the second phalanx with the Leonardo board mounted on the back of the hand. The goal is to evaluate the performance in the following scenarios: when no finger is already in contact and that our algorithm has to determine which of the five fingers enters in contact. In addition we include 2-finger chords: tapping with two fingers simultaneously on the touchpad (from the user perspective) when no fingers are already in contact. Considering our algorithm uses the highest voltage within time windows, simultaneous contacts reduce chances to correctly identify fingers.

The participant sat in front of a desktop computer and the prototype was equipped on the hand they reported to use for operating a pointing device. They were then invited to interact either with a KEYBOARD or TOUCHPAD in response to a visual stimuli. The stimuli displayed an image of a hand with a circular overlay on the finger or chord they had to use (Fig. 1-right). It also displayed the name of the finger or chord in plain text. Participants then had to operate the corresponding device with the requested finger or chord. Once the interaction had been performed, there was a 400 ms delay before displaying the next stimuli to avoid participants' anticipation.

For the keyboard part of the experiment, participants had to perform two different types of *operations* on the keyboard: a TYPE which corresponds to typing a key, starting with the finger not in contact with the key, and a PUSH which corresponds to pressing a key with the finger first positioned on it. We also used 2 different *keyboards*: an *Apple Magic Keyboard*, which has LOW travel distance keys, and a *Hewlett-Packard KU-0316*, which has HIGH travel distance keys. Only the 5 individual FINGERS were tested in this part. The participant could press on any letter key of the keyboard. If more than one key was pressed, the trial had to be repeated.

For the touchpad part of the experiment, participants had to perform two different types of *operations*: a TAP which corresponds to tapping the touchpad, and a SLIDE which corresponds to performing a quick unidirectional movement in any direction right after the finger enters in contact with the touchpad. There was no requirement for the direction of the movement so that we have no prior knowledge when doing the recognition. Participants could touch the touchpad at any location, but had to repeat the trial if more than the required number of touches was detected. The 5 individual FINGERS as well as the 10 possible 2-fingers CHORDS were tested. Participants used an external Apple Magic Trackpad.

To avoid influencing participants, no particular instructions on the hand position were given and no feedback informed the participants on the detected finger(s) after each trial.

For each trial, we logged the expected finger or chord and the actual finger or chord detected by the *simple* algorithm. We video recorded all the sessions that we manually annotated with the actual fingers used by the participants. Raw data was displayed over the video for easier gesture labeling.

### Design

Half of the participants started with the keyboard part of the experiment, the other half starting with the touchpad part.

The keyboard part of the experiment used a $2 \times 2 \times 5$ within-subject design with factors *keyboard* (LOW or HIGH travel distance keys), *operation* (TYPE or PUSH), and *contacts* (all 5 FINGERS of the dominant hand). Each combination of these factors was repeated 5 times, for a total of $20 \times 2 \times 2 \times 5 \times 5 = 2000$ trials for all participants. Orders of *Keyboard* and *operation* were counter-balanced across participants.

The touchpad part of the experiment used a $2 \times 15$ within-subject design with factors *operation* (TAP or SLIDE) and *contacts* (5 FINGERS + 10 CHORDS)[1]. Each combination of these factors was repeated 5 times, resulting in a total of $20 \times 2 \times 15 \times 5 = 3000$ trials for all participants. Order of *operation* was counter-balanced across participants.

The order of *contacts* was randomized with the only constraint that a contact cannot appear more than twice in a row.

[1] CONTACTS can also be replaced by the *number of contacts* (1 or 2 fingers) that can also be considered as a factor.

| **Keyboard** | All operations | TYPE | PUSH |
|---|---|---|---|
| All keyboards | 98.2%<br>1965 / 2000 | 98.3%<br>983 / 1000 | 98.2%<br>982 / 1000 |
| LOW | 99.1%<br>991 / 1000 | 98.4%<br>492 / 500 | 99.8%<br>499 / 500 |
| HIGH | 97.4%<br>974 / 1000 | 98.2%<br>491 / 500 | 96.6%<br>483 / 500 |
| **Touchpad** | All operations | TAP | SLIDE |
| CONTACTS | 96.3%<br>2890 / 3000 | 96.9%<br>1454 / 1500 | 95.7%<br>1436 / 1500 |
| FINGERS | 99.7%<br>997 / 1000 | 99.6%<br>498 / 500 | 99.8%<br>499 / 500 |
| CHORDS/ SETS | 94.7%<br>1893 / 2000 | 95.6%<br>956 / 1000 | 93.7%<br>937 / 1000 |

**Table 1: Detailed recognition rates of WhichFingers for the keyboards and touchpad.**

### Results

In the subsequent analysis, we used SPSS for the ANOVAs. Mauchly tests indicated the assumption of sphericity was not violated in any of our analysis.

For the keyboard part, repeated measures ANOVA only found a significant main effect of *keyboard* ($F_{1,19} = 7.5$, $p < 0.02$, $\eta_p^2 = 0.28$) on recognition rate. Overall WhichFingers successfully identified 99.1% of fingers on the LOW keyboard and 97.4% on the HIGH keyboard (see Table 1 for details).

For the touchpad part, repeated measures ANOVA only found a significant main effect of *contacts* ($F_{14,266} = 8.3$, $p < 0.0001$, $\eta_p^2 = 0.30$) and *number of contacts* ($F_{1,19} = 30.4$, $p < 0.001$, $\eta_p^2 = 0.62$) on recognition rate. Overall WhichFingers successfully identified 96.3% of the contacts: 99.7% of the FINGERS, 94.7% of the CHORDS (right set of fingers and correct contact-finger assignment) and 94.7% of the SETS (right set of fingers and wrong contact-finger assignment). Pairwise comparisons between the different *contacts* showed many significant differences between them (significant differences are highlighted in Fig. 2). The Middle-Little chord (83%) performs significantly worse than all other chords except Thumb-Little. Thumb-Little with 90.5% is the second worse chord and performs significantly worse than the other ones except Index-Little, Middle-Little and Thumb-Ring.

## EXPERIMENT WITHOUT CONSTRAINT ON FINGERS

In the first experiment we systematically investigated the recognition rates of fingers and chords. We also wanted to evaluate WhichFingers in tasks representative of daily activities on a desktop computer while not constraining the fingers or chords used by the participants.

### Method

In this experiment, we asked 12 participants (28.3 mean age, all right-handed, 1 female, computer science university staff and students) to complete 4 tasks with either the *touchpad* or the *keyboard*, on a desktop computer. See also the accompanying video for task demonstrations. Participants used the same trackpad as the first experiment and the Apple Magic Keyboard.

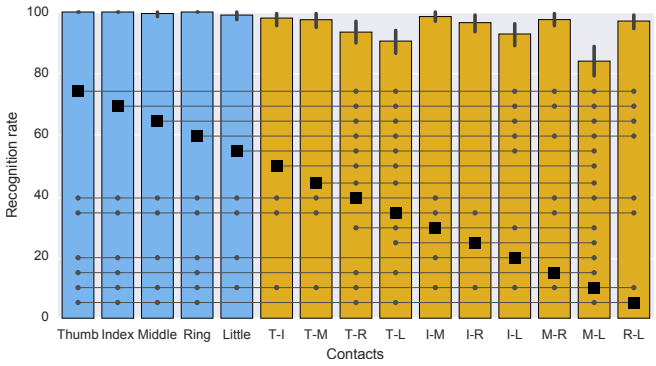


**Figure 2: Mean success rates and 95% CI for the touchpad per *contact*. TAP and SLIDE are merged. The horizontal lines represent the significant differences between *contacts*: for a given line, the square *contact* is significantly different from the dot *contacts*.**

*Touchpad.* Participants were equipped with the device on their dominant hand and had to perform 3 distinct *tasks* using the touchpad: a docking task, a scrolling task and a pointing task. For the docking task, the interface displayed a geometric shape that the participant had to scale, rotate and position in its similar shaped dock by only using gestures on the touchpad. The transformation initially applied to the shape was generated randomly. The only available operations were 2-finger rotate gestures for rotating the shape, 2-finger pinch-and-expand gestures for scaling the shape, and dragging for positioning the shape. For the scrolling task, the interface displayed at the top of a window a target word that the user had to acquire. The target was contained in a 4254 line, alphabetically-ordered list that was displayed in a 29-lines-high viewport. The participants had to scroll the list using 2-finger scrolling gestures only and acquired the target by clicking on it. For the pointing task, participants had to click on a 1.4 cm wide circular target randomly positioned 28.2 cm away. After selection, the participant had to lift all fingers from the touchpad to display the next target. Each of these tasks consisted of 12 trials. In total, these tasks lasted about 6 minutes. During these tasks, we collected the data retrieved from the device, as well as the current task and all the low-level inputs performed on the touchpad (contact points information, button clicks).

*Keyboard.* Participants were equipped with the device on their left hand and had to perform a text entry and formatting task using the keyboard only in Apple Pages with its toolbar hidden. The view was configured in full screen, with the leftmost part displaying Apple Pages and the rightmost part displaying a non-editable version of the text participants had to type and format, as well as the formatting command names and their corresponding keyboard shortcuts below the text so participants did not have to learn and memorize them. Participants had to use the keyboard only. Text had to be selected using a combination of the shift modifier and the arrow keys. Commands had to be selected using their corresponding keyboard shortcuts, that had been modified so participants could perform them using the left hand only. They were allowed to use both hands for typing text and selecting commands. We equipped the wired version of the device on participants' left hands because we expected participants to issue keyboard

| | | | 2-fingers | |
|---|---|---|---|---|
| | CONTACTS | FINGERS | CHORDS | SETS |
| Touchpad | 94.6% | 98.2% | 91.9% | 93% |
| | 3165 / 3346 | 1394 / 1419 | 1771 / 1927 | 1792 / 1927 |
| Keyboard | 99.2% | 99.4% | 98% | 98% |
| | 2993 / 3017 | 2542 / 2557 | 451 / 460 | 451 / 460 |

**Table 2: Recognition rates of WhichFingers for the second experiment.**

shortcuts (and especially to press modifier keys) mostly with the left hand. In total, the task lasted approximately 10 minutes. During this task, we collected data from the device, as well as the low-level key events.

### Results

We logged a total of 3,346 gestures on touchpad (1,419 one-finger contacts and 1,927 two-finger contacts) and 3,017 on keyboard (2,557 one-key presses and 460 two-key presses). We video recorded all trials and used the videos to manually label each contact point of the gestures with the finger used by the participants. WhichFingers successfully identified 94.6% of the CONTACTS performed on touchpad: 98.2% of the FINGERS and 91.9% of the CHORDS. Note the recognition rate increases to 93% for SETS. For the key presses on keyboard, WhichFingers successfully identified 99.2% of them: 99.4% of the one-key presses and 98% of the two-key presses (*e.g.* the ones using key modifiers). Note there was no wrong key/finger assignments for the keyboard with two-key presses. Table 2 summarizes the detailed recognition rates.

## INTERACTION TECHNIQUES EXAMPLES

WhichFingers allows designers to replicate interaction techniques previously developed using other prototypes, but also allows them to explore new interaction techniques. We explored interaction techniques for multi-touch trackpads on desktop computers as finger identification has never been used in this context. The techniques we developed, illustrated in the accompanying video, run on macOS using the Apple Cocoa API. To work system wide, all the implemented applications run in the background and overwrite the OS pointing, scrolling and window management system.

The first technique we implemented is a pointing technique combining relative and absolute pointing. When dealing with large screens and/or a large number of them, cursor displacements can be tedious, especially when the current cursor position is far away from the target. The idea is to use one finger for relative cursor positioning and another one for absolute cursor positioning by mapping the trackpad surface to the screens. The user can first roughly position the cursor at the desired position using absolute pointing before using relative pointing to precisely select a target. Malacria *et al*. observed that the index and the middle fingers are the most common used fingers for pointing on touchpads [20]. We therefore kept those two fingers for relative pointing and used the ring finger (almost never used for pointing) for absolute pointing. This idea is similar to ARC-Pad but using finger identification for mode switching [23].

The second technique we implemented is a scrolling technique similar to the previous idea. Scrolling long documents is often performed with two fingers on a touchpad and can result in numerous motor actions in order to position the view port on a given page. We propose to use the index and middle fingers for relative scrolling and the middle and ring fingers for absolute scrolling by mapping the whole document to the touchpad height. As for pointing, the idea is to roughly position the viewport using absolute scrolling before adjusting precisely with relative scrolling.

Lastly, we implemented a window management system that ease the positioning of windows side by side. A common feature in current OS is to resize a window to one half of the available screen real estate (most of the time by dragging it to one side of the screen). We mimic that behaviour using fingers as modifiers: while dragging a window using the index finger, the user can specify to resize it on the left part of the screen by tapping with the thumb, on the right part of the screen by tapping with the little finger and maximize the window by tapping with both.

## DISCUSSION

### Single contacts accuracy

The results of our two experiments first highlight that WhichFingers is an accurate low-latency solution to identify individual fingers in various conditions.

For the keyboards the recognition rate is 98.2% on average in the first experiment and 99.2% in the second one. The first experiment did not show an effect of the way of operating the keys (fingers first positioned on or above the keys) but a significant effect of keyboard with the HP keyboard presenting high travel distance keys (97.4%) compared to the Apple keyboard having low travel distance keys (99.1%). We hypothesis this small difference is due to a softer spring effect with the HP keyboard that dampens the contacts. For the touchpad WhichFingers gets very good results to identify a single finger. The recognition rate is 99.7% in the first experiment and 98.2% in the second one. Simply tapping with the finger on the surface or tap and sliding did not affect the technique. Overall these results are similar or above results previously published in the literature in contexts that were more constrained or restricted compared to ours [15, 2].

More generally these results indicate that fingers entered in sequence, one after another, can be detected in a robust way. The same applies when the two hands are equipped and a single finger of each hand is input, even simultaneously.

### Two or more simultaneous contacts accuracy

Given that our algorithm is based on the time matching between input events and peaks generated by the piezo sensors, events occurring within a short time window (30 ms in our algorithm) are more likely to be wrongly identified. Recognizing two finger chords is indeed less robust compared to single finger contacts with 94.7% recognition rate in the first experiment and 91.9% in the second one. Note that the finger sets recognition rate was similar to the chords rate in the first experiment (because the participants did not rotate their hand when performing the gesture), but higher than the chords rate (93% compared to 91.9%) in the second experiment.

Considering current interaction techniques explored in the literature [9, 11, 13, 14, 20, 29], being able to correctly identify a chord when fingers are entered simultaneously has never been explored in the literature while being able to correctly identify which fingers are in contact is more useful and present higher success rates.

### Causes of recognition errors

The detailed analysis of the errors in the two experiments revealed different causes of errors. First in some situations the wires collide with each others which produces a signal on a neighboring sensor that is then wrongly interpreted. Our prototype could be improved by using less rigid wires to prevent such problems or move the sensor on the third phalanx near the palm instead of the second phalanx. Also using wireless communication between the sensors and the board handling sensors' data would solve this problem.

The first experiment shows that some chords, corresponding to hand postures that are difficult to articulate, get lower rates. We hypothesize users tend to tense their hand in order to hold difficult postures making the sensitivity of all the vibration sensors increase. We also noticed for the keyboard, the index finger touched the thumb's sensor because the thumb was just beneath the index finger (on the space bar). Moving the sensor closer to the palm would help solve this problem.

For the second experiment where participants mainly used chords they are used to perform (*e.g.* thumb+index, index+middle), the lower success rates can be explained by the use of rotational movements in the experiment: when the time elapsed between two contacts is less than 30 ms our algorithm uses the x position of the contacts, which can be oversimplistic in such situations, for instance when a user starts a rotate gesture with inverted thumb and index fingers in order to increase the range of movement.

Last with very light presses on the touchpad the response of the sensors is not high enough and may be misinterpreted with the noise of the other sensors. Reducing the noise of the signal would help solve this problem.

### Improving accuracy

We designed our hardware and software to make them easy to replicate. We therefore intentionally used a simple signal processing algorithm. The simple algorithm can be used as the backbone on which heuristics tailored to a particular problem can be added. Further improvements would depend on the interaction context, and would need to be tailored to each application. For example on a touchpad when inputting a chord using the index and middle finger with the right hand, the left contact is expected to be an index finger. On a keyboard one can expect the numerical keypad cannot be used with the left hand. Since the interaction context matters, we wanted to assess first the "raw" accuracy of the device with simple heuristics tailored for our contexts. In addition, after testing the device, we believe future work should focus on improving the signal quality and accuracy (precise latency estimation and high frequency signal) rather than delegating the signal processing to a machine learning algorithm.

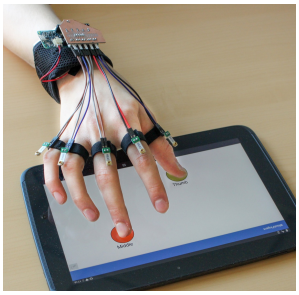

**Figure 3: Our wireless device prototype. The micro-controller and battery are attached to the wrist. The sensors are moved to the third phalanx near the palm.**

### How designers would use WhichFingers

WhichFingers is mainly targeted at HCI researchers who want to explore interaction and conduct performance studies with techniques leveraging finger identification. Most interaction techniques using finger identification that have been proposed in the literature can easily be re-implemented using our prototype. However these applications are rarely compared to existing techniques. Proving the benefit of finger identification without reliable and responsive hardware remains challenging since the most accepted arguments involve time comparisons. The finger identification rates and low-latency of our technology make such comparisons possible.

Although WhichFingers is easily reproducible and software examples are available[2], the current version of the system still requires designers to implement ad-hoc applications that fuse low level data from both our device and the operating system events (*e.g.* key events or touch events) and tune the time-frame window parameters. Indeed, synchronization between the device and the host computer is a critical factor as the success rate can be affected by the jitter on latency in the communication. Finally, we want to stress again that the benefits of WhichFingers come at the cost of equipping users hand(s) but, depending on the envisioned application, not all fingers need to be instrumented.

### Wireless version

We developed WhichFingersWiFi, a second prototype using an Arduino MKR1000 attached to a bracelet using Velcro, to process the raw information and send the identified fingers over WiFi to the host computer (Fig. 3). The simplicity of our algorithm makes it possible to run on the micro-controller. When an event is produced on the host-computer it is sent to the micro-controller that analyzes the voltages over the time window and reports the identified fingers to the host-computer. In total, this version costs less than $50. Future iterations of WhichFingersWiFi could use wireless rings as demonstrated by Fukumoto *et al.* [7, 8].

With this version, the sensors are positioned on the third phalanx to solve some of the problems aforementioned. We did not observe any significant difference in term of finger identification. Carefully examining the influence of these positions, in terms of comfort and performance, is left for future work.

[2] All the necessary material to replicate the device (EAGLE files and Arduino code), data collected as well as software used are available at `http://ns.inria.fr/mjolnir/whichfingers`

## CONCLUSION

We presented WhichFingers, a robust, low latency and low cost device that relies on vibration sensors attached to each finger to identify them. Through two experiments we have demonstrated that WhichFingers shows high recognition rates for keyboards and touch surfaces. After dozens of device prototypes developed over the last decade to identify fingers, WhichFingers is the first alternative presenting good results in various scenarios to achieve low-latency identification of the five fingers of the hand and all the 2-finger chords, making it an applicable solution for HCI researchers willing to evaluate the performance of interaction techniques using finger identification.

## REFERENCES


1. Oscar Kin-Chung Au and Chiew-Lan Tai. 2010. Multitouch Finger Registration and Its Applications. In *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction (OZCHI '10)*. ACM, New York, NY, USA, 41–48. DOI: **http://dx.doi.org/10.1145/1952222.1952233**
2. 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 (ITS '09)*. ACM, New York, NY, USA, 93–100. DOI: **http://dx.doi.org/10.1145/1731903.1731924**
3. Géry Casiez, Thomas Pietrzak, Damien Marchal, Sébastien Poulmane, Mathieu Falce, and Nicolas Roussel. 2017. Characterizing Latency in Touch and Button-Equipped Interactive Systems. In *Proceedings of UIST'17, the 30th ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 9. DOI: **http://dx.doi.org/10.1145/3126594.3126606**
4. Ashley Colley and Jonna Häkkilä. 2014. Exploring Finger Specific Touch Screen Interaction for Mobile Phone User Interfaces. In *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: The Future of Design (OzCHI '14)*. ACM, New York, NY, USA, 539–548. DOI: **http://dx.doi.org/10.1145/2686612.2686699**
5. Philipp Ewerling, Alexander Kulik, and Bernd Froehlich. 2012. Finger and Hand Detection for Multi-touch Interfaces Based on Maximally Stable Extremal Regions. In *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces (ITS '12)*. ACM, New York, NY, USA, 173–182. DOI: **http://dx.doi.org/10.1145/2396636.2396663**
6. A. Fadell, A. Hodge, S. Schell, R. Caballero, J.L. Dorogusker, S. Zadesky, and E. Sanford. 2014. Embedded authentication systems in an electronic device. (July 15 2014). **https://www.google.com/patents/US8782775** US Patent 8,782,775.
7. Masaaki Fukumoto and Yasuhito Suenaga. 1994. "FingeRing": A Full-time Wearable Interface. In *Conference Companion on Human Factors in Computing Systems (CHI '94)*. ACM, New York, NY, USA, 81–82. DOI: **http://dx.doi.org/10.1145/259963.260056**
8. Masaaki Fukumoto and Yoshinobu Tonomura. 1997. "Body Coupled FingerRing": Wireless Wearable Keyboard. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '97)*. ACM, New York, NY, USA, 147–154. DOI: **http://dx.doi.org/10.1145/258549.258636**
9. Emilien Ghomi, Stéphane Huot, Olivier Bau, Michel Beaudouin-Lafon, and Wendy E. Mackay. 2013. Arpège: Learning Multitouch Chord Gestures Vocabularies. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces (ITS '13)*. ACM, New York, NY, USA, 209–218. DOI: **http://dx.doi.org/10.1145/2512349.2512795**
10. Alix Goguey, Géry Casiez, Thomas Pietrzak, Daniel Vogel, and Nicolas Roussel. 2014a. Adoiraccourcix: Multi-touch Command Selection Using Finger Identification. In *Proceedings of the 26th Conference on L'Interaction Homme-Machine (IHM '14)*. ACM, New York, NY, USA, 28–37. DOI: **http://dx.doi.org/10.1145/2670444.2670446**
11. Alix Goguey, Géry Casiez, Daniel Vogel, Fanny Chevalier, Thomas Pietrzak, and Nicolas Roussel. 2014b. A Three-step Interaction Pattern for Improving Discoverability in Finger Identification Techniques. In *Proceedings of the Adjunct Publication of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST'14 Adjunct)*. ACM, New York, NY, USA, 33–34. DOI: **http://dx.doi.org/10.1145/2658779.2659100**
12. Alix Goguey, Mathieu Nancel, Géry Casiez, and Daniel Vogel. 2016. The Performance and Preference of Different Fingers and Chords for Pointing, Dragging, and Object Transformation. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4250–4261. DOI: **http://dx.doi.org/10.1145/2858036.2858194**
13. Alix Goguey, Daniel Vogel, Fanny Chevalier, Thomas Pietrzak, Nicolas Roussel, and Géry Casiez. 2017. Leveraging finger identification to integrate multi-touch command selection and parameter manipulation. In *IJHCS Journal*. Elsevier, 21–36. DOI: **http://dx.doi.org/10.1016/j.ijhcs.2016.11.002**
14. Aakar Gupta, Muhammed Anwar, and Ravin Balakrishnan. 2016. Porous Interfaces for Small Screen Multitasking Using Finger Identification. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 145–156. DOI: **http://dx.doi.org/10.1145/2984511.2984557**

15. Aakar Gupta and Ravin Balakrishnan. 2016. DualKey: Miniature Screen Text Entry via Finger Identification. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 59–70. DOI: **http://dx.doi.org/10.1145/2858036.2858052**

16. 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 (UIST '13)*. ACM, New York, NY, USA, 41–50. DOI: **http://dx.doi.org/10.1145/2501988.2502021**

17. Noor A Ibraheem. 2016. Finger Identification and Gesture Recognition Using Gaussian Classifier Model. In *IJAER journal*. Research India Publications, 6924–6931. **http://www.ripublication.com/ijaer16/ijaerv11n10_08.pdf**

18. G. Julian Lepinski, Tovi Grossman, and George Fitzmaurice. 2010. The Design and Evaluation of Multitouch Marking Menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 2233–2242. DOI: **http://dx.doi.org/10.1145/1753326.1753663**

19. K. Ma, J. Li, J. Han, Y. Hu, and W. Xu. 2016. Enhancing touchscreen input via finger identification. In *Proc. CSTIC*. IEEE, 1–3. DOI: **http://dx.doi.org/10.1109/CSTIC.2016.7463927**

20. Sylvain Malacria, Alix Goguey, Gilles Bailly, and Géry Casiez. 2016. Multi-touch Trackpads in the Wild. In *Actes De La 28e Conférence Francophone Sur L'Interaction Homme-Machine (IHM '16)*. ACM, New York, NY, USA, 19–24. DOI: **http://dx.doi.org/10.1145/3004107.3004113**

21. Shahzad Malik, Abhishek Ranjan, and Ravin Balakrishnan. 2005. Interacting with Large Displays from a Distance with Vision-tracked Multi-finger Gestural Input. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology (UIST '05)*. ACM, New York, NY, USA, 43–52. DOI: **http://dx.doi.org/10.1145/1095034.1095042**

22. Nicolai Marquardt, Johannes Kiemer, David Ledo, Sebastian Boring, and Saul Greenberg. 2011. Designing User-, Hand-, and Handpart-aware Tabletop Interactions with the TouchID Toolkit. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS '11)*. ACM, New York, NY, USA, 21–30. DOI: **http://dx.doi.org/10.1145/2076354.2076358**

23. David C. McCallum and Pourang Irani. 2009. ARC-Pad: Absolute+Relative Cursor Positioning for Large Displays with a Mobile Touchscreen. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 153–156. DOI: **http://dx.doi.org/10.1145/1622176.1622205**

24. Sundar Murugappan, Vinayak, Niklas Elmqvist, and Karthik Ramani. 2012. Extended Multitouch: Recovering Touch Posture and Differentiating Users Using a Depth Camera. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 487–496. DOI: **http://dx.doi.org/10.1145/2380116.2380177**

25. R.M. Schmitt and R.A. Craig. 2014. Biometric sensing device for three dimensional imaging of subcutaneous structures embedded within finger tissue. (Aug. 7 2014). **https://www.google.com/patents/US20140219521** US Patent App. 14/174,761.

26. Measurement Specialties. 2017. Minisense 100 Vibration sensors. (2017). **http://www.te.com/usa-en/product-CAT-PFS0011.html**, visited July 10th, 2017.

27. Atsushi Sugiura and Yoshiyuki Koseki. 1998. A User Interface Using Fingerprint Recognition: Holding Commands and Data Objects on Fingers. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology (UIST '98)*. ACM, 71–79. DOI: **http://dx.doi.org/10.1145/288392.288575**

28. Katia Vega and Hugo Fuks. 2013. Beauty Tech Nails: Interactive Technology at Your Fingertips. In *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction (TEI '14)*. ACM, New York, NY, USA, 61–64. DOI: **http://dx.doi.org/10.1145/2540930.2540961**

29. Julie Wagner, Eric Lecolinet, and Ted Selker. 2014. Multi-finger Chords for Hand-held Tablets: Recognizable and Memorable. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2883–2892. DOI: **http://dx.doi.org/10.1145/2556288.2556958**

30. Jingtao Wang and John Canny. 2004. FingerSense: Augmenting Expressiveness to Physical Pushing Button by Fingertip Identification. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems (CHI EA '04)*. ACM, New York, NY, USA, 1267–1270. DOI: **http://dx.doi.org/10.1145/985921.986040**

31. Wayne Westerman. 1999. *Hand tracking, finger identification, and chordic manipulation on a multi-touch surface*. Ph.D. Dissertation. University of Delaware. **http://www.eecis.udel.edu/~westerma/main.pdf**

32. M. Yousefpor, J.M. Bussat, B.B. Lyon, G. Gozzini, S.P. Hotelling, and D. Setlak. 2015. Fingerprint Sensor in an Electronic Device. (2015). **http://www.google.com/patents/US20150036065** US Patent App. 14/451,076.

33. Jingjie Zheng and Daniel Vogel. 2016. Finger-Aware Shortcuts. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4274–4285. DOI: **http://dx.doi.org/10.1145/2858036.2858355**