Extending Keyboard Shortcuts with Arm and Wrist Rotation Gestures

Daniel Buschek[1], **Bianka Roppelt**[1], **Florian Alt**[1,2]
[1]LMU Munich, [2]Munich University of Applied Sciences; Munich, Germany
{daniel.buschek, florian.alt}@ifi.lmu.de, bianka.roppelt@campus.lmu.de

ABSTRACT
We propose and evaluate a novel interaction technique to enhance physical keyboard shortcuts with arm and wrist rotation gestures, performed during keypresses: rolling the wrist, rotating the arm/wrist, and lifting it. This extends the set of shortcuts from key combinations (e.g. *ctrl + v*) to combinations of key(s) and gesture (e.g. *v + roll left*) and enables continuous control. We implement this approach for isolated single keypresses, using inertial sensors of a smartwatch. We investigate key aspects in three studies: 1) rotation flexibility per keystroke finger, 2) rotation control, and 3) user-defined gesture shortcuts. As a use case, we employ our technique in a painting application and assess user experience. Overall, results show that arm and wrist rotations during keystrokes can be used for interaction, yet challenges remain for integration into practical applications. We discuss recommendations for applications and ideas for future research.

ACM Classification Keywords
H.5.2. Information Interfaces and Presentation (e.g. HCI): Input devices and strategies (e.g. mouse, touchscreen)

Author Keywords
Keyboard; smartwatch; wrist rotation; elicitation study

INTRODUCTION
Mechanical keyboards are an important part of the user interface for desktop computers and laptops. Besides text entry, they are used to provide shortcuts ("hotkeys") in many applications, such as browsing, gaming, image manipulation, and 3D modelling. Such shortcuts are fast and convenient for frequently used actions [7]; they reduce mouse movements and keyboard-mouse switches and need no screen space.

However, shortcuts usually only offer discrete one-directional control. For example, increasing and decreasing volume in a music player app would require two different shortcuts, although users might think of both as shortcuts to the same command and/or control element (i.e. volume dial/slider).

We propose and evaluate a novel interaction technique to enhance physical keyboard shortcuts with arm and wrist rotation gestures. These are performed while holding a key pressed down with one finger. As a starting point, this paper investigates isolated single keypresses, not shorcuts interleaved with typing or rotations while pressing multiple keys. Our gestures enable continuous control in multiple directions, akin to a joystick. For example, changing the volume can be assigned to one key with gestures (e.g. *v + roll wrist left/right*).

Figure 1. We propose to extend keyboard shortcuts with arm and wrist rotations gestures, performed while pressing down a key. The figure shows left/right wrist rolls. The user's rotation angles can be used, for example, for continuous control, such as changing the volume.

We implement this approach with a smartwatch and sensor data from its inertial measurement unit (IMU). We evaluate this concept and prototype in three user studies on: 1) rotation flexibility per keystroke finger, 2) rotation control, and 3) user-defined gesture shortcuts. We also assess user experience in a use case with a painting application.

We reveal opportunities and challenges: Users can control arm and wrist rotations while pressing an (isolated) key to reach and hold target angles within useful limits. Maximum reachable angles depend on hand posture, finger and direction. Abduction is more flexible and convenient than adduction. Rolling the wrist is the favourite gesture, lowering the arm the least preferred one. However, rotation shortcuts took longer and had higher workload than normal ones in our use case study, revealing the challenges of integrating the technique into practical applications. Assigning shortcuts themselves, people varied in their mappings yet particularly associated gestures with matching directional and dichotomous commands.

Overall, investigating arm and wrist rotations as a way of extending keyboard shortcuts, this paper contributes: 1) A concept for combining keyboard interaction with arm and wrist rotations; 2) its implementation in a protoype system with example application, using off-the-shelf devices; and 3) an in-depth evaluation in several user studies.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada.

ACM 978-1-4503-5620-6/18/04...$15.00
https://doi.org/10.1145/3173574.3173595

RELATED WORK

Many concepts enrich keyboard or touch input. We focus on those useful for shortcuts. Table 1 presents an overview.

Taylor et al. [15] augmented a mechanical keyboard with infrared proximity sensors in between the keys to enable different types of gestures: swiping and drawing shapes on the keyboard, multi-touch gestures, and (micro-)hover gestures above it. In contrast, we use off-the-shelf devices without modifying the keyboard. Conceptually, our rotation gestures require contact with the keyboard (i.e. pressing a key), yet also use behaviour above it. Their participants suggested combining keypresses with continuous control, motivating our work.

The "Air+Touch" concept by Chen et al. [5] also combines both contact and hover elements. They mounted a depth-camera to a smartphone to enable mid-air gestures executed *before*, *between*, or *after* screen touches (e.g. circling for continuous zoom, "high jump" for text selection). In their design space, our gestures present another category, namely gesturing *during* contact with the screen or, in our case, the keyboard.

Wilkinson et al. [16] proposed this combination for interaction with a tablet. Their *Expressy* approach uses a wrist-worn inertial sensor to enrich touch input. Like our concept, they include wrist rotations. *Touch+* by Hwang et al. [11] shows a similar combination of handheld touch with smartwatch IMU sensors. These projects on mobile touchscreens motivate our exploration of a similar concept for physical keyboards.

Also enriching touch, *TapSense* by Harrison et al. [8] uses tapping sounds on touchscreens to distinguish finger parts. *FingerSense*[1] brings this to mobile devices. The approach could enhance shortcuts (e.g. same key triggers *copy* with tip, but *cut* with knuckle). Instead of sound, smartphone IMUs were used by Hinckley and Song [9] to enrich mobile touch input (e.g. *hold-and-shake*), while Goel et al. [6] used them to infer pressure and hand posture. *Fat Thumb* by Boring et al. [4] uses touch area as an extra degree of freedom. Instead of implicit touch sounds, device grip and movements, or touch area, we distinguish users' controlled arm/wrist rotations while pressing physical keys.

Bailly et al. [2] modified individual keys of a mechanical keyboard for their *Métamorphe* concept. These keys can be lowered and raised and users can twist and push them in four directions. They propose shortcut gestures (e.g. *f* + *push left/right* for "find previous/next"). This motivates our similar use of directional associations between commands and physical controls. However, we explore this idea for arm/wrist rotation gestures, not "key gestures". Hence, we use a wrist-worn IMU (smartwatch) instead of a modified keyboard.

GestKeyboard by Zhang et al. [19] enables gestures on unmodified mechanical keyboards: Users drag the finger across multiple keys. This is separated from typing with spatio-temporal features (e.g. proximity of subsequently pressed keys). On average, gestures used 10.2 keys. In contrast, we map functions to one (main) key each, with gestures added via rotation of arm and wrist, not dragging across other keys.

[1] http://www.qeexo.com/

what	when	where	control	off-the-shelf	input device	reference
chorded hotkeys	at key down/up	contact	discrete	yes	keyboard	Au et al. [1]
actuated movable keys	during keystroke	contact	continuous, discrete	no	keyboard	Bailly et al. [2]
touch contact area	at/during touch	contact	continuous, discrete	yes	smartphone	Boring et al. [4]
touch + mid-air gestures	before/ after tap	contact/ above, sequential	continuous, discrete	no	smartphone	Chen et al. [5]
pressure & posture via IMU/vibr.	at/during touch	contact	continuous, discrete	yes	smartphone	Goel et al. [6]
touch with finger parts	at touch down	contact	discrete	no	smartphone	Harrison et al. [8]
touch + motion via IMU	during touch	contact	discrete	yes	smartphone	Hinckley & Song [9]
key pressure via IMU	at keystroke	contact	continuous, discrete	yes (laptop with IMU)	laptop keyboard	Iwasaki et al. [12]
gestures on/above keyboard	keystroke indepen-dent	contact or above	continuous, discrete	no	keyboard	Taylor et al. [15]
touch + wrist IMU	before/ during tap	contact and above	continuous, discrete	yes	tablet	Wilkinson et al. [16]
touch + finger orientation	during tap	contact and above	continuous, discrete	yes (mod. kernel)	smartphone	e.g. [13, 14, 18]
gestures across keys	across keystrokes	contact	discrete	yes	keyboard	Zhang et al. [19]
finger-specific hotkeys	at keystroke	contact	discrete	no	keyboard	Zheng et al. [20]
keypress + arm/wrist rotation	during keystroke	contact and above	continuous, discrete	yes	keyboard	*this paper*

Table 1. A (non-exhaustive) overview of concepts useful for extending shortcut options and expressiveness of virtual/physical buttons/keys. Regarding the listed dimensions, we are the first to explore continuous control during keypresses on an unmodified physical keyboard.

Au et al. [1] proposed keyboard shortcuts via simultaneously pressing/releasing multiple keys ("chords"). They do not require modifier keys (e.g. *ctrl*) to distinguish them from typing. Instead of timing thresholds for chords, our concept enables rotation thresholds. Rotations could then serve a similar purpose (e.g. shortcut: key + rotation; typing: key only).

Zheng et al. [20] presented "finger-aware" shortcuts, another method to distinguish shortcuts and typing: Keys react differently based on finger, hand, and posture used to press them. Their evaluation shows that performance and convenience depend on the used finger. This motivates our comparison of wrist rotation gestures for keystroke with different fingers.

Iwasaki et al. [12] used accelerometers in laptops to sense key pressure for expressive typing, for example, to modify font size. While pressure might also be estimate with a wrist-worn IMU [16], we focus on rotation gestures in this paper.

Related work inferred finger orientation from capacitive sensor data for rich touchscreen interaction [13, 14, 18]. This motivates our work on utilising orientation and rotation of arm and wrist. These are connected (kinematic chain), but not the same as finger orientation.

In summary, related work has proposed a rich set of augmentations for physical keyboards and touchscreens and their use for shortcuts. Many used modified keyboards with extra sensors. This makes these ideas difficult to disseminate with current hardware. Moreover, none of these projects used arm/wrist rotations to enhance physical keyboards. This motivates our investigation of this novel combination in this paper.

COMBINING KEYBOARDS WITH ARM/WRIST ROTATIONS

Motivation

We see three main benefits of adding arm and wrist rotation gestures to key presses (cf. [2]).

Increasing expressiveness: Enabling gestures while pressing keys extends the design space for keyboard interactions. In our concept users can turn each key into a simple "finger joystick". This increases the expressiveness of keyboard input.

More single-key shortcuts: Adding gestures extends the number of possible shortcuts, in particular for single keys. For example, an application might assign both "rotate left" and "rotate right" commands to the "r"-key, distinguished by left and right wrist rotations, respectively.

Continuous control: Gestures extend shortcuts from discrete triggers to continuous controls. For example, a music player application might allow users to increase/decrease the volume by pressing a key and lifting the arm up/down, respectively.

Concept and Approach

Our general concept in this paper is to use arm and wrist rotation gestures during keypresses to extend physical keyboard interactions. This concept could be enabled in a variety of ways. Here we decided to use off-the-shelf devices, combining an unmodified keyboard with a smartwatch. The latter provides the IMU to sense rotations.

By choosing this approach for implementing our concept, we sense rotation of one arm, namely the non-dominant one, on which users wear the watch. We argue that this is suitable for our target use case in this paper: extending keyboard shortcuts in typical interaction setups with the dominant hand on the mouse and the other one on the keyboard (e.g. 3D modelling, graphics software). However, our concept itself does not exclude the use of rotations of both arms and wrists.

Implementation

We used an LG G Watch R connected via Bluetooth to an LG Nexus 5. The phone communicated with a MacBook Pro via websockets. Custom Android and web apps allow applications to request sensor data from the phone, which reads these from the watch and forwards them to the laptop. In particular, we use the *Game Rotation Sensor* from the Android API, which yields relative rotation changes. However, their ranges vary for the different dimensions; hence we map them to a standard 0 to 360 degree range for better understanding and comparisons in our analyses. Moreover, our software computes rotation changes relative to a starting value. In our concept, this is the hand posture at the moment of the key down event.

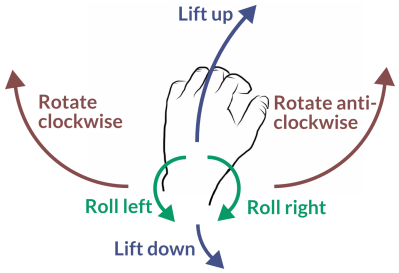

Figure 2. The six rotation directions.

OVERVIEW: RESEARCH APPROACH

We conducted three focussed user studies on key aspects of our concept, plus a use case study: First, we evaluate per keypress finger how far users can rotate their arm and wrist while holding down the key, starting in a typing posture. Second, we evaluate rotations in a control task, regarding behaviour, speed, and perceived workload. The third study assesses user-defined shortcuts. Finally, we integrate our interaction technique into a painting application, to study a practical use case.

STUDY I: ROTATION FLEXIBILITY PER FINGER

The goal of this first study was to assess the maximum arm and wrist rotation angles that users could reach while pressing a key. Knowing these limits is important, for example, to inform mappings for interaction techniques.

Study Design

Our within-subject design had two independent variables: 1) *finger*, with one level per finger/thumb of the left hand; 2) rotation *direction*, with six levels, see Figure 2. As the dependent variable, we measured the maximum reachable rotation angle.

Participants

We recruited twelve people (eight female) via colleagues and friends. Their mean age was 34 years (range: 11 - 55). One was left-handed. The mean hand size was 18.0 cm (range: 16.1 - 20.8), measured between index finger tip and lunate bone.

Apparatus

We developed a simple Processing[2] application that displayed the current study condition (finger, direction). It ran on a laptop placed on a table. While pressing any key, it recorded sensor data from a smartwatch that people wore on their left wrist. The relative location of user and keyboard was controlled, that is, people sat at the table directly in front ot the laptop.

Procedure

We invited participants to our lab, explained our idea and study and demonstrated the gestures. We asked them to assume their normal resting posture as if they were typing. In each task, they had to press the spacebar with the displayed finger of their left hand and rotate their wrist in the displayed direction, *as far as possible* without releasing the key. People could restart tasks. Task order was counterbalanced with a Latin square. While the spacebar offers some flexibility in keypress location, people consistently hit the left third, due to left hand use and the controlled setup.

[2] https://processing.org/

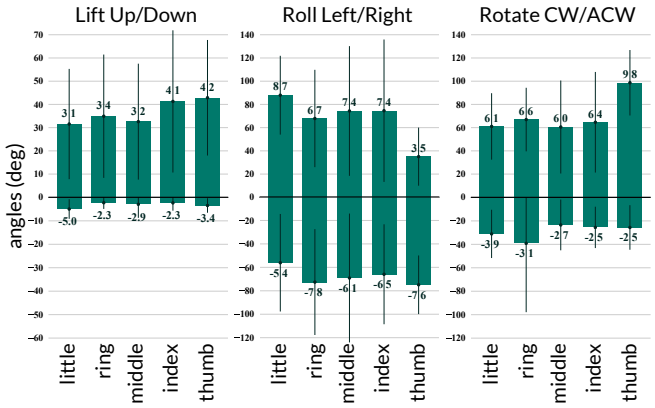

Figure 3. The mean furthest reachable rotation angles per finger for the six rotation directions.

Results and Discussion

We recorded 360 hand movements (12 participants × 6 directions × 5 fingers). Figure 3 shows the results.

User variability: High standard deviations reveal the user-specific nature of these reachable angles. We observed several influencing factors: hand size, finger nail length, injury (one person had had wrist surgery), and rotation strategy (e.g. for lift: bending the wrist vs lifting the arm). Moreover, we observed some key slips for users trying to reach extreme angles.

Comparison of gestures: The typing posture does not leave much space for lifting the arm down. Moreover, the arm allows for greater abduction (clockwise rotation) than adduction (anticlockwise). Rolling the wrist is most flexible overall.

Comparison of fingers: Figure 3 shows that limits vary between fingers, as is to be expected. For example, when pressing a key with the left thumb, rolling the wrist to the left is hindered by the fingers. Similarly, rolling to the right is difficult while pressing the key with the little finger.

Based on our results, we selected the middle finger as a robust and neutral choice for further exploration in the next study.

STUDY II: ROTATION CONTROL

The goal of this second study was to further assess rotation behaviour and workload in a control task.

Study Design

We used a within-subject design with two independent variables: 1) rotation *direction*, as in the first study (see Figure 2); 2) *target angle*, with five levels for each of these directions (see *x*-axes in Figure 5), informed by the ranges observed in the first study: We set the ranges to cover the sum of the absolute min/max values per dimension – this is an estimate of the total range including posture preparations. This allowed us to observe if/when/how people would modify their keystroke posture (e.g. starting with raised arm for far lift down tasks). Hence, we did not prescribe a certain hand posture in this study. We measured IMU data as timeseries, task completion time, and workload with the *NASA TLX* questionnaire.

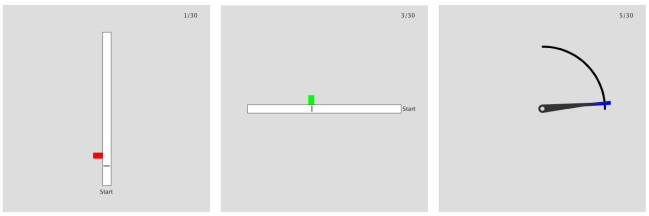

Figure 4. The three control elements used in the second study, from left to right: lift up, roll left, rotate clockwise. The same elements with inverted starting locations were used for the opposite directions. The screenshots also show the three target states: not reached (red), reached (green), and completed, that is, held for two seconds (blue).

Participants

We recruited twelve participants (six female) via colleagues and friends with a mean age of 35 years (range: 20 - 61). Two were left-handed. Their mean hand size was 18.8 cm (range: 16.6 - 20.9), measured as in the first study. Participants were compensated with 7.50 €.

Apparatus

Figure 4 shows the study application, again implemented with Processing. For each task, it displayed a control element indicating current and target values. The target indicator changed its colour depending on the task status (see procedure below). The target width covered 4 % of the total range.

Procedure

We invited people to our lab, explained the study and demonstrated the gestures. In each task, they pressed down the spacebar with the middle finger of their left hand and used the given rotation axis to bring the indicator into the target area (see Figure 4). The target changed its colour from red to green once the user reached it, and further to blue, once the user held the target value for two seconds. Releasing the key in the blue state completed the task and started the next one. In contrast, releasing the key prematurely was counted as a failure and the same task was repeated until success. After each task, participants filled in a *NASA TLX* questionnaire. Task order was counterbalanced according to a Latin square.

Results

Our data contains 360 successful tasks (12 participants × 6 directions × 5 target angles). In addition, there are 17 failed tasks of which we excluded four for our analyses since they resulted from accidental key presses, not real attempts.

Time: Figure 5 shows the times of successful tasks without holding time. The grand mean was 3.66 seconds (SD 2.04 s). ANOVA showed a significant effect of both *direction* ($F(5,55) = 3.45$, $p < 0.01$, $\eta^2 = 0.24$) and *target angle* ($F(4,44) = 18.42$, $p < 0.001$, $\eta^2 = 0.63$) on time. The interaction was also significant ($F(20,220) = 2.54$, $p < 0.001$, $\eta^2 = 0.19$). Bonferroni-corrected post-hoc tests revealed that anticlockwise arm rotation was significantly slower than all other directions (apart from clockwise rotation), and that all angle differences were significant (apart from second vs third angle, and fourth vs fifth), all $p < 0.05$. Overall, farther rotations took longer.

Failure cases: On average, people failed 3.61 % of the tasks (SD 4.80 %). Over 58 % of participants never failed at all over the 30 tasks. Almost half of all failures occurred for the anti-clockwise arm rotation. In contrast, lifting the arm never failed. A third of all failures occurred for the largest target angle, a quarter for the second largest, indicating that more extreme rotations are more difficult. We did not observe accidental finger lifts (i.e. releasing the key during interaction) and people did not mention any.

Leaves: We define a "leave" as bringing the indicator into the target area but leaving again before holding it for two seconds (e.g. overshooting). The grand mean of "leaves" per task was 1.13 (SD 0.66). More "leaves" happened for lift up/down than for the others, and more extreme angles had a higher number.

Workload: Figure 5 shows workload as rated by participants via *NASA TLX*. We observed lowest workloads for lift up/down and highest for rotate (anti-)clockwise. ANOVA showed a significant effect of both *direction* ($F(5,55) = 7.00$, $p < 0.001$, $\eta^2 = 0.39$) and *target angle* ($F(4,44) = 11.02$, $p < 0.001$, $\eta^2 = 0.50$) on workload. The interaction was also significant ($F(20,220) = 2.31$, $p < 0.005$, $\eta^2 = 0.17$). Bonferroni-corrected post-hoc tests revealed that lift up/down had significantly lower workload than rotating the arm (anti-)clockwise. Anticlockwise rotation also had significantly higher workload than all other directions. Angles showed two groups: the two shortest ones had significantly lower workload than the two highest ones (all $p < 0.05$). Furthermore, perceived workload was significantly correlated with both the number of "leaves" ($r = 0.18, p < 0.01$) and time ($r = 0.22, p < 0.01$).

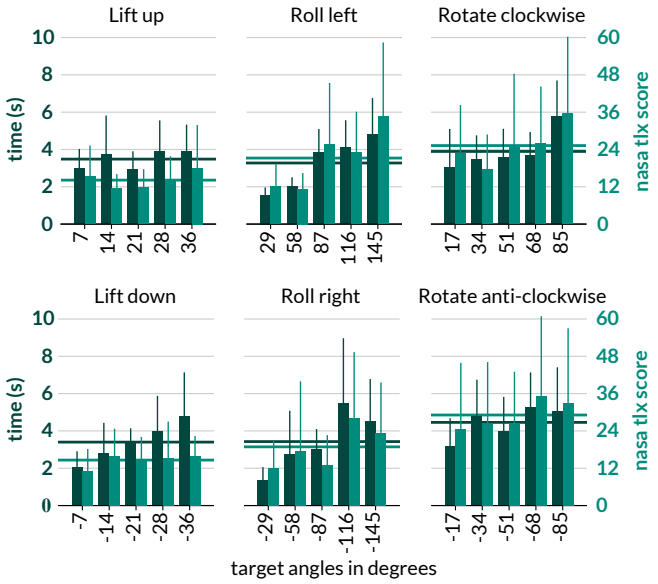

Figure 5. Times (without holding time) and *NASA TLX* scores per task. Horizontal bars indicate the means per rotation direction.

Discussion

The results shed light on control aspects of arm and wrist rotation gestures for keyboard shortcuts.

Reaching Rotation Angles

The majority of people succeeded in most tasks on the first try. They were free to choose their starting angle and mostly pressed the key in a position that allowed them to perform the required rotation. A few failures happened due to an inadequate starting posture (e.g. arm already down for lift down). In these cases, second attempts were successful, indicating a learning effect. Some people also commented that gestures felt easier at the end compared to the start of the study.

More time was needed to reach more extreme angles, as is to be expected. Rotating the arm (anti-)clockwise was slowest overall. People's comments indicate that anticlockwise rotation (adduction) in particular was cumbersome, matching the workload ratings and the relatively limited angles observed in the first study.

In contrast, lifting was fastest and rated best overall, also for larger angles. Relatively low workload scores were also assigned to short left/right wrist rolls (<60 degrees).

Holding Rotation Angles

Staying close to a target rotation angle was easier for some people than others. The timeseries data revealed overshooting. This happened more for more extreme angles and in particular for lift up/down. The latter suggests that the used range for lifting could be refined. More generally, future work could explore different mappings. Nevertheless, participants were able to correct their postures in cases of overshooting, successfully completing all tasks.

Conclusion

We conclude that arm and wrist rotation angles can be controlled well within reasonable limits while pressing a key on a keyboard. However, for larger angles, users need to prepare their keystroke hand postures to have enough flexibility for the rotation movement. While people seemed to get used to this in our study, applications may favour gestures with shorter movements. They should also expect overshooting and prefer rolling the wrist left/right and lifting the arm, compared to rotating it; adduction in particular should be avoided.

Based on our results, we recommend to use the following angle ranges (in degrees, left hand): lift down/up -36 to 36; roll right/left -87 to 116; and rotate anti/clockwise -51 to 68.

STUDY III: USER-DEFINED GESTURE SHORTCUTS

In a survey, we investigated how users would utilise the gesture-extended shortcut vocabulary. Note that it is not our goal to use this to replace designers, but to assess how people relate gestures to commands "upon first contact".

We follow similar methodologies employed by Wobbrock et al. [17] and Bailly et al. [2]: Participants assign shortcuts to commands. To do so, they are presented with the *effects* of the command in a short video (e.g. screen capture of zooming in on an image). Participants then have to indicate the input that they think should lead to this result. Like Bailly et al. [2], we informed participants about the full set of commands at the start of the study, to facilitate globally suitable groupings and avoid "local optimisations".

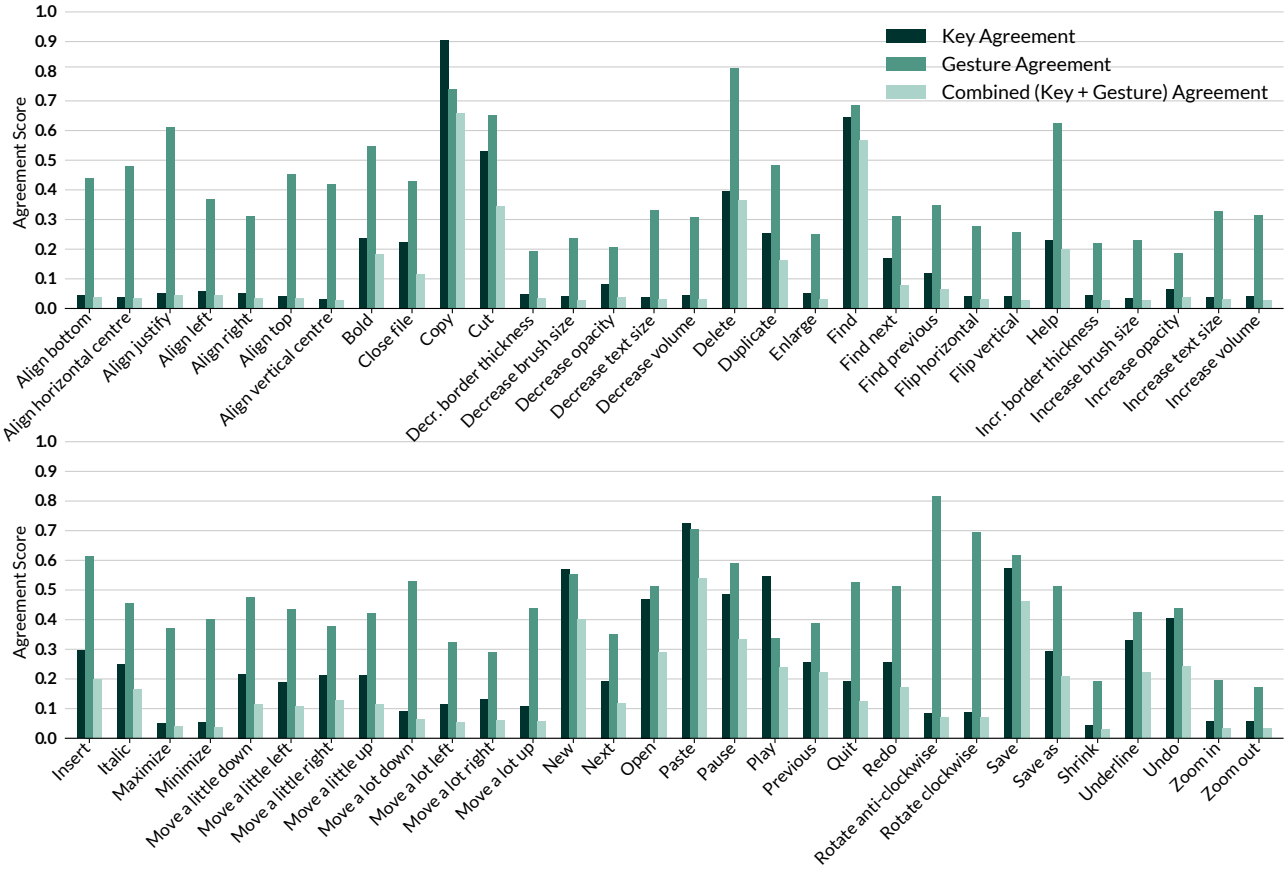

Figure 6. Agreement scores. Note that high gesture agreement does not imply that a gesture was perceived as useful (i.e. high agreement on "no gesture").

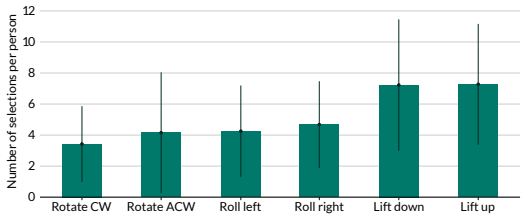

Figure 7. Gestures in shortcut assignments in the elicitation survey.

Participants

We advertised our questionnaire via social media and a university newsletter. It was completed by 40 people (15 female) with a mean age of 25 years (range: 17 - 55). Most were students (>85 %), mostly related to computer science (>66 %). Everyone who completed the questionnaire had a chance of winning one of three 20 € gift cards.

Apparatus and Procedure

We ran this study as an online survey. It explained the technique with video demonstrations of the six rotational directions (Figure 2). The survey also displayed an illustration of a keyboard, plus a list with all 60 commands. The commands were based on those from Bailly et al. [2], with a few refinements and additions. The commands can be found in Figure 6.

After the introduction, the survey showed one page per command in random order. On each page, a video presented the effects of the command. Shortcuts were assigned by selecting up to three keys and/or a gesture from dropdown menus.

We did not enforce the use of gestures to be able to assess for which commands users would find them useful. Hence, it was possible to choose "no gesture". In addition, participants rated the usefulness of a gesture for each command on a five-point Likert scale. Each page also showed the keyboard illustration again to make users aware of all possible keys.

Results

Our data contains 2400 shortcut assignments (40 participants $\times$ 60 commands).

Keys: Averaged across participants, 31.58 (SD 4.07) different keys were used across all commands. Shortcuts consisted of an average of 1.88 keys (SD 0.67). One of the modifier keys *ctrl* and *alt* was present in 61.25 % of shortcut assignments.

Gestures: Figure 7 shows the distribution of gestures. On average, people assigned gestures to 29 commands (SD 10.96) out of 60. They used 5.65 (SD 0.91) of the six possible gestures; 34 of 40 people used all six gestures at least once. Lift up/down were most popular. When people assigned gestures to both commands in a dichotomous pair (e.g. shrink/enlarge, previous/next), they selected opposite gesture directions (e.g. roll left/right) in 72.67 % (SD 22.91 %) of cases.

Agreement scores: We computed agreement scores as defined by Wobbrock et al. [17]. Following Bailly et al. [2], we use three types: agreement on keys, on gestures, and combined agreement. Figure 6 shows these scores per command. Overall, highest key agreement occurred for the most common

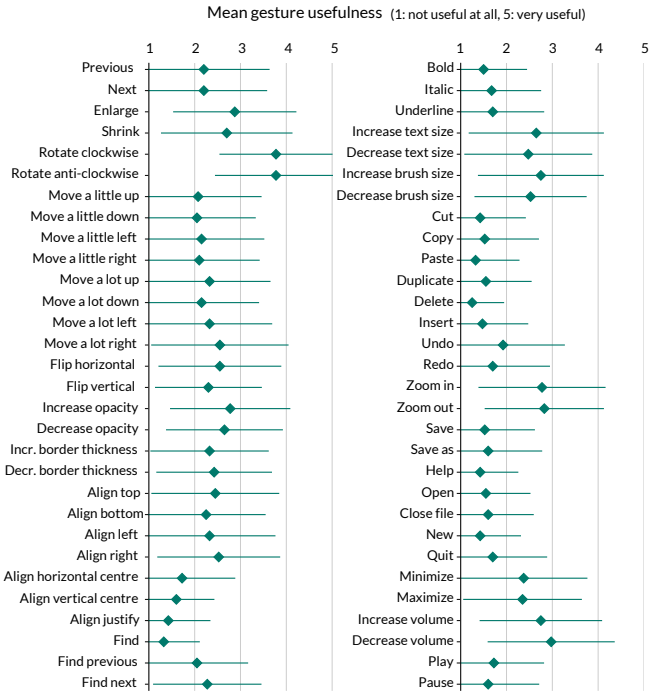

Figure 8. Summary of Likert ratings on perceived usefulness of using gestures in a shortcut for each command.

commands with well-known mnemonic shortcuts (e.g. *ctrl+c* for *copy*). Highest gesture agreement (apart from "no gesture" cases) occurred for directional commands (e.g. *rotate* object).

Gesture usefulness: Figure 8 summarises the Likert ratings on gesture usefulness per command. Overall, participants found gestures to be most useful for rotating objects, zooming, and changing values such as volume, text size, and brush size.

Discussion

Most people considered all available gestures and used them selectively, yet in a considerable proportion of assignments (ca. 50 %). Well-known modifier keys were used in over half of the assignments. Perceived usefulness of gestures was low in general, yet clearly spiked for commands which inherently contain a direction. In these cases, participants assigned shortcuts with matching gestures. Bailly et al. [2] made a similar observation for their physically movable keys.

Our findings suggest that people did not easily "give up" well-known shortcuts. They only used gestures where they perceived them as valuable extensions. We conclude that applications should integrate gestures in line with existing shortcuts. They have the particular opportunity to augment those commands which are semantically related to directional effects.

We found a considerable spread of answers, indicated by low agreement on gestures for most commands (Figure 6) and varied ratings on usefulness (Figure 8). While this might indicate that there is some flexibility in possible gesture assignments, it also raises questions on discoverability and learnability. Our results thus suggest that applications should actively support users in discovering and learning gesture assignments, for example with feedforward techniques, and/or by limiting gesture

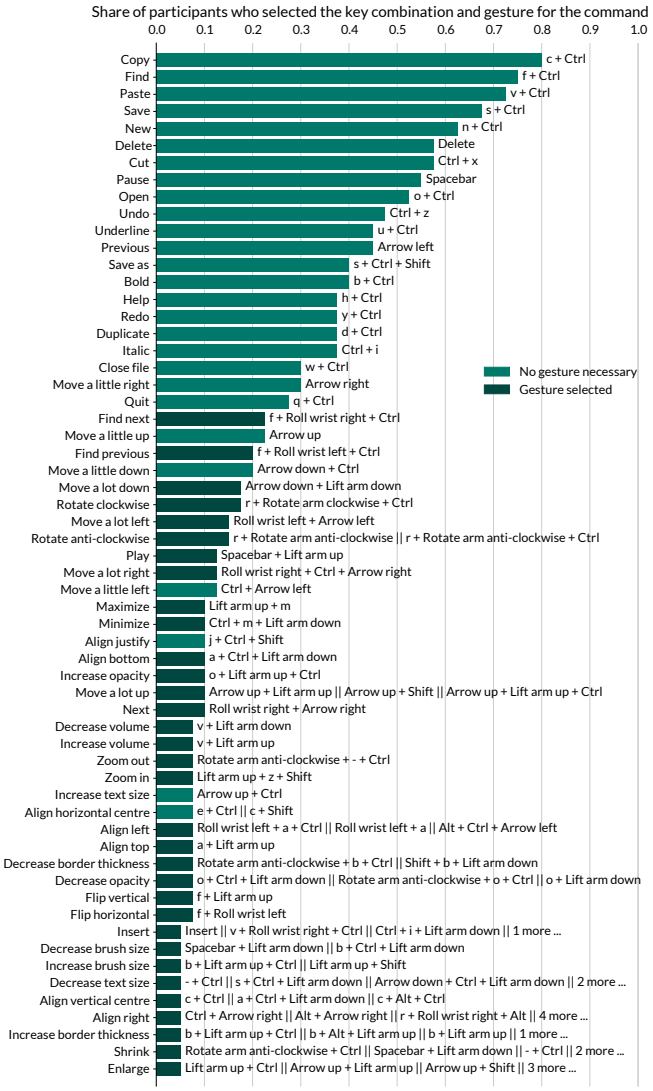

Figure 9. The user-defined set of shortcuts, ordered by agreement score.

options (e.g. only for directional commands and/or only one dimension such as wrist rolls).

The top user-defined shortcuts are shown in Figure 9. Additional aspects should be considered for implementation: For example, pressing two keys with the same hand likely makes gestures more difficult to perform. Thus, modifier keypresses might only be required initially, not held during gestures.

USE CASE: PAINTING APPLICATION

As a practical use case, we conducted a user study with a painting software to assess user experience with our interaction technique and prototype in an application context.

Study Design

We used a within-subject design: Each participant used our drawing application both with normal shortcuts and with added gestures. We logged all interactions, including timestamps, current settings, and so on. We also assessed workload and user experience with *NASA TLX* and *AttrakDiff* questionnaires.

Participants
We recruited 21 participants (twelve female) via social media and a university newsletter. They had a mean age of 25 years (range: 18 - 54). Most were students (>80 %). One person was left-handed. Participants were compensated with either 12.50 € or study credits. Their median experience with painting programs was four on a five-point Likert scale.

Apparatus
We implemented a custom drawing and painting application in Processing (Figure 10). It was controlled with mouse and keyboard, plus the gestures. It included typical tools like drawing with the mouse, inserting text and geometric shapes, plus scaling, rotating, translating, and aligning them. Colour and line width could be adjusted as well.

For the gesture version, we created a command mapping based on the user-defined mapping from the elicitation study. However, we did not take it directly but rather refined it manually for consistency for our application. For shortcuts with multiple keys, users only had to keep one key pressed during the gesture. This was motivated by greater flexibility for hand postures, choice of fingers/keys, and rotations.

In addition, we printed two "cheat sheets" for the shortcuts; one for the normal set, the other for the set with gestures.

Procedure
We invited participants to our lab and introduced them to our drawing software. They used it freely for ten minutes to get familiar with it. We then introduced gestures and again gave them ten minutes to test the software with the new shortcuts.

After the introduction, participants followed a list of tasks (e.g. "create a large vertical ellipse"), that resulted in a simple painting. Participants then followed another list of tasks to create a second painting. The task list was placed as a printout next to the participant; it was also read aloud by the experimenter, with additional explanations in case of questions.

One painting was created with normal shortcuts, the other one with gestures. Any keystroke finger could be used. The two paintings and their subtasks were not identical but equivalent (head of bear and cat, see Figure 11). The order of the normal and gesture conditions and the assignment of paintings to these conditions was counterbalanced with a Latin square.

After completing each painting, participants filled in *NASA TLX* and *AttrakDiff* questionnaires to measure workload and perceived attractiveness of the interaction technique (normal vs gestures). The study then concluded with a short interview.

Results
Task Completion Time
Figure 12 shows the completion times with standard shortcuts and with gestures for the different tasks. We found a significant effect of both *task* ($F(2.57,51.45) = 14.64$, $p < 0.001$, $\eta^2 = 0.42$) and *method* ($F(1,20) = 42.85$, $p < 0.001$, $\eta^2 = 0.68$) on time. The interaction was also significant ($F(2.65,53.07) = 4.63$, $p < 0.01$, $\eta^2 = 0.19$). Participants took significantly longer with gestures than without ($p < 0.001$).

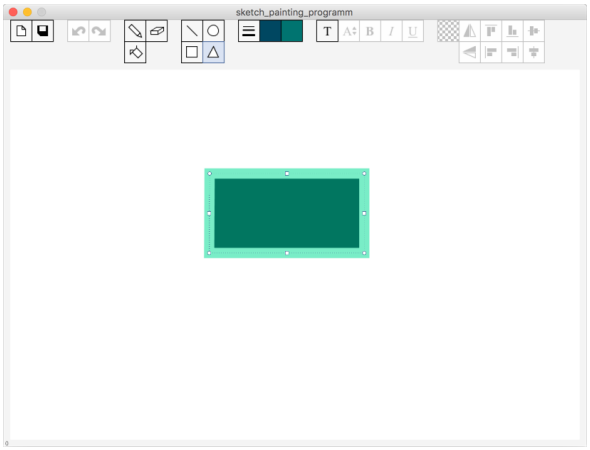

Figure 10. The painting program with toolbar and canvas, used to test the proposed gesture shortcuts in an application context.

Figure 11. The two study paintings (bear, cat), each defined by a list of tasks. This figure shows concrete results; the first two columns were created with gestures, the last column without.

Accuracy
Several subtasks required changing a slider value (for line weight, opacity, text size). We measured how accurately people matched the target values. We found no significant differences in accuracy between the standard interaction and using gestures ($t(20) = -0.75$, $p = 0.46$).

Workload
Creating the painting with gestures resulted in a mean *NASA TLX* score of 49.98 (SD 16.88). The version without gestures was rated with a mean of 21.49 (SD 9.84). The difference was significant ($t(20) = -7.41$, $p < 0.001$).

User Experience
Figure 13 summarises the *AttrakDiff* results. The prototype with gestures had a higher hedonic quality, whereas the one without gestures was perceived as more pragmatic.

Additionally, we asked users to rate the convenience of the six gestures as well as ease-of-use when employing them for each task (Figure 14). Lift down was perceived as least convenient, while best ratings were given to roll right and clockwise arm rotation. Highest ratings of ease-of-use were observed for gesture shortcuts related to 1) settings with a slider (border thickness, opacity, text size) and 2) moving objects on the canvas. In contrast, controlling the colour picker (cursor on 2D colour space) was rated more difficult.

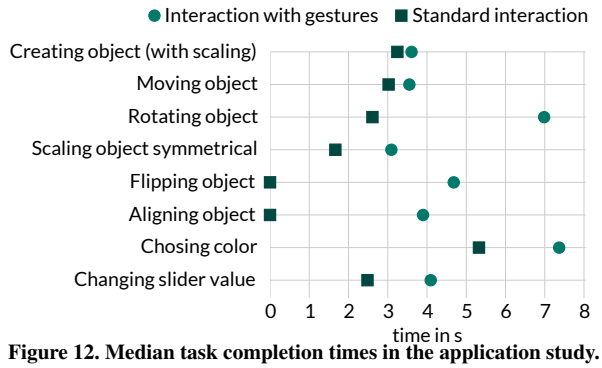

Figure 12. Median task completion times in the application study.

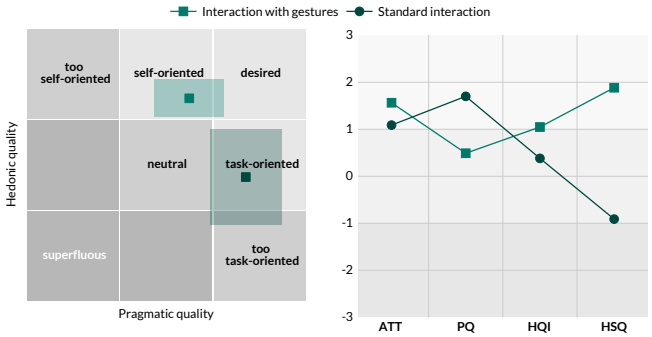

Figure 13. Summary of users' ratings in the *AttrakDiff* questionnaire.

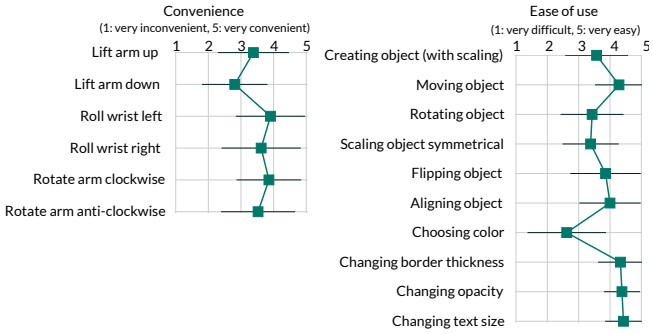

Figure 14. Left: Ratings on convenience of performing the gestures as part of the painting application. Right: Ratings on ease-of-use of employing these gestures for each task.

Feedback in the post-study interviews matched these results: Using gestures for slider settings was highlighted as intuitive and easy to use, fitting to the assignments with high gesture agreement in the elicitation study. However, participants found it difficult to control two dimensions simultaneously (e.g. colour picker). Overall, the concept was well received and most people said they could imagine using it with an improved implementation; many also mentioned playful applications.

LIMITATIONS

Our implementation approach requires a smartwach (or another wrist-worn IMU [16]). When sensed in this way, rotation gestures can only be used for one hand. This fits their use for shortcuts quite well, considering a typical setup with the dominant hand on the mouse and the other one on the keyboard. However, other use-cases may require different sensors to assess rotations related to both hands.

We did not study the use of these gestures interlaced with typing, yet based on our observations typing does not involve (considerable) arm and wrist rotations similar to our gestures. Moreover, while our painting application included shortcuts with multiple keys, users only had to keep one key pressed during rotations. Hence, we did not study gestures for multiple keys held at once (key chords).

We do not claim to have identified the ideal processing of sensor data and our lab studies could not cover long-term everyday use. While our prototype was useful in this first exploration, we will improve it for long-term use and evaluations in applications. Related, we only tested one smartwatch and a laptop keyboard. Different watches (and sensor placements therein), as well as raised keys might change (sensed) behaviour details. Conceptually, we only studied position control; rate control could be investigated as an alternative.

While we measured maximum angles and related workload, a future study could explicitly investigate which (maximum) angles are most *comfortable* for the users.

DISCUSSION AND IMPLICATIONS

Interaction with Arm/Wrist Rotations During Keystrokes

Promisingly, our work shows that arm and wrist rotations while pressing a (single) key on a keyboard can be utilised for interaction. This offers several rotation directions per key with useful ranges, also for different fingers. Users can control these rotation movements to reach and hold desired values. Our prototype system and application demonstrate that this interaction technique can be enabled for unmodified keyboards using the sensors of a smartwatch.

Challenges for Practical Applications

Direct comparison for our application shows that controlling it with gestures achieves comparable accuracy and painting results, yet takes longer than without them. This also reflects in higher ratings on pragmatic quality for the non-gesture version. We discuss these issues regarding several aspects.

First, our *mapping* was not ideal. Our application used gestures for (almost) every command so that we could test them intensively. As our elicitation study indicates, a real application might better integrate gestures by using them more selectively.

In particular, gestures were more practical for some *tasks* than others: For example, 2D control was difficult and simple tasks with discrete standard commands (e.g. flip object) should not require a gesture, since they do not benefit from continuous control. In contrast, moving an object and changing other continuous (slider) values seems more promising (higher subjective ratings, smaller time difference).

In this regard, the role of *feedback* needs further investigation. For some tasks, sliders provided continuous feedback that matched the rotation direction (e.g. opacity), in contrast to others (e.g. choose colour). The ones with such matching feedback received better ratings. Some participants also explicitly noted that this match of visuals and gesture directions contributed to better ease-of-use. Also taking into account the results of the second study (which also had feedback), these findings thus motivate further studies on continuous feedback.

Finally, the *prototype* could be tuned further, also towards practical integration, for example by using a direct link from watch to PC, instead of sending data via the phone. Separating concept and prototype, future work could also investigate other sensing methods (e.g. external sensors, motion tracking).

Recommendations on Gesture Directions

We studied six rotation directions (Figure 2). Our results point towards a set of recommendations for applications:

1) Lowering the arm should be avoided. Although in an abstract task it was fast and its workload was rather low, this gesture has little reach, especially without adjusting the keystroke posture in advance (i.e. pressing the key with a raised arm). Participants also rated this as the least convenient gesture in our application study, matching their comments during use.

2) If not all directions are needed, applications should favour abduction (away from body; here: roll left, rotate clockwise) over adduction (towards body; here: roll right, rotate anticlockwise). Our results show that this allows for larger movements. Abduction gestures also received slightly better ratings than their adduction counterparts in the application study. Participants' feedback throughout all studies clearly confirms this.

3) If an application needs just one gesture type, we recommend rolling left/right. This gesture was most flexible across fingers and directions. It was also fast and received lowest workload ratings for short angles in the control study. Most people also indicated it as their favourite gesture in the application study.

Avoid Requiring Long Rotations

Some combinations of keystroke fingers and rotation directions result in shorter movement ranges (Figure 3). For larger angles, users might also need to modify the hand posture with which they press the key to leave enough room to move. Larger angles also take more time, require larger movements, and thus result in higher workload. Moreover, observations and feedback from our last study indicate that users can perform gestures accurately as part of an application task without requiring overly large rotation angles. Based on our results, we thus recommend to limit the required rotation angles.

Use Gestures for Directional & Dichotomous Commands

User-defined mappings showed high gesture agreement for directional commands. People also assigned fitting gesture pairs to dichotomous commands. In the painting application study, gestures received high ratings on ease-of-use for such directional and dichotomous commands. Participants also said in the interviews that sliders were easy and intuitive to control with gestures. We observed comparable accuracy with and without gestures for these commands. Hence, our studies suggest that (pairs of) gestures are most intuitive and easy to use when associated with (dichotomous) directional commands.

Towards Other Use Cases

To bring the technique to other use cases, several extensions are required, in particular for gesture shortcuts interleaved with other keyboard use such as text entry. We see the following steps for future work: 1) study naturally occurring arm/wrist rotations during typing, 2) measure ranges and control for other finger/key combinations, 3) extend the system to distinguish gestures from normal typing movements (our prototype did not require this, since we had no other keyboard task besides the gestures). Our results provide starting points: For example, it is unlikely that less flexible directions in our tasks – or angles requiring prepared postures – will be suitable choices in more constrained postures while typing.

CONCLUSION

We proposed and evaluated arm and wrist rotation gestures to extend keyboard interaction. These gestures are performed while pressing a key. They extend the set of possible shortcuts and enable continuous control along multiple axes. Thus, our concept increases expressiveness of keyboard interaction.

We implemented this concept for isolated single keypresses, using smartwatch sensors, and investigated four key aspects: 1) rotation flexibility per keystroke finger, 2) rotation control, 3) user-defined gesture shortcuts, and 4) user experience. We found that users can control rotations with useful accuracy and ranges for interaction. However, we also revealed challenges: In particular, rotation shortcuts did not outperform normal ones in our case study and elicited assignments indicated that gesture mappings might not be obvious without supporting users in discovering and learning them. Nevertheless, overall we conclude that arm and wrist rotations during keystrokes can be used to extend keyboard interaction and shortcuts, and offer a promising direction for future work.

FUTURE WORK

As a main next step, the technique could be investigated for more general shortcuts: 1) gestures while pressing multiple keys (key chord shortcuts), and 2) gesture shortcuts interleaved with other keypresses (especially typing). Future work could also focus on further utilising the continuous control aspect. For example, in a painting application such as ours, the gestures could enable users to change line width *dynamically* while drawing with the mouse.

We see rich further opportunities: Rotation gestures could be used for gaming, acting like joysticks, or to modify the "strength" of an action. Another area is authentication: Combined with characters, rotation gestures increase the password space. They could also lead to novel authentication secrets which combine textual and graphical passwords (e.g. press certain keys interlaced with certain gesture drawings).

Introducing these gestures could be improved with feedforward techniques [3]. Moreover, detecting the keystroke finger [10] would enable finger-specific rotation mappings and shortcuts. Conceptually, we focussed on gestures while pressing a key; this could be extended to mid-air (rotation) gestures above the keyboard, before or after a keystroke (cf. [5]).

ACKNOWLEDGEMENTS

Work on this project was partially funded by the Bavarian State Ministry of Education, Science and the Arts in the framework of the Centre Digitisation.Bavaria (ZD.B). This research was supported by the Deutsche Forschungsgemeinschaft (DFG), Grant No. AL 1899/2-1.

REFERENCES

1. Oscar Kin-Chung Au, Kira Yip-Wo Yeung, and Jacky Kit Cheung. 2016. DownChord and UpChord: A New Style of Keyboard Shortcuts Based on Simultaneous Key-down and Key-up Events. In *Proceedings of the Fourth International Symposium on Chinese CHI (ChineseCHI2016)*. ACM, New York, NY, USA, Article 3, 10 pages. DOI: http://dx.doi.org/10.1145/2948708.2948715

2. Gilles Bailly, Thomas Pietrzak, Jonathan Deber, and Daniel J. Wigdor. 2013. Métamorphe: Augmenting Hotkey Usage with Actuated Keys. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 563–572. DOI:http://dx.doi.org/10.1145/2470654.2470734

3. Olivier Bau and Wendy E. Mackay. 2008. OctoPocus: A Dynamic Guide for Learning Gesture-based Command Sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST '08)*. ACM, New York, NY, USA, 37–46. DOI: http://dx.doi.org/10.1145/1449715.1449724

4. Sebastian Boring, David Ledo, Xiang 'Anthony' Chen, Nicolai Marquardt, Anthony Tang, and Saul Greenberg. 2012. The Fat Thumb: Using the Thumb's Contact Size for Single-handed Mobile Interaction. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '12)*. ACM, New York, NY, USA, 39–48. DOI:http://dx.doi.org/10.1145/2371574.2371582

5. Xiang 'Anthony' Chen, Julia Schwarz, Chris Harrison, Jennifer Mankoff, and Scott E. Hudson. 2014. Air+Touch: Interweaving Touch & In-air Gestures. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 519–525. DOI: http://dx.doi.org/10.1145/2642918.2647392

6. Mayank Goel, Alex Jansen, Travis Mandel, Shwetak N. Patel, and Jacob O. Wobbrock. 2013. ContextType: Using Hand Posture Information to Improve Mobile Touch Screen Text Entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 2795–2798. DOI: http://dx.doi.org/10.1145/2470654.2481386

7. Tovi Grossman, Pierre Dragicevic, and Ravin Balakrishnan. 2007. Strategies for Accelerating On-line Learning of Hotkeys. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. ACM, New York, NY, USA, 1591–1600. DOI: http://dx.doi.org/10.1145/1240624.1240865

8. Chris Harrison, Julia Schwarz, and Scott E. Hudson. 2011. TapSense: Enhancing Finger Interaction on Touch Surfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 627–636. DOI: http://dx.doi.org/10.1145/2047196.2047279

9. Ken Hinckley and Hyunyoung Song. 2011. Sensor Synaesthesia: Touch in Motion, and Motion in Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 801–810. DOI: http://dx.doi.org/10.1145/1978942.1979059

10. 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

11. Sungjae Hwang, John Song, and Junghyeon Gim. 2015. Touch+: Expanding Touch Input Vocabulary Using a Smartphone and a Smartwatch. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 165–165. DOI: http://dx.doi.org/10.1145/2702613.2732485

12. Ken Iwasaki, Takashi Miyaki, and Jun Rekimoto. 2009. Expressive Typing: A New Way to Sense Typing Pressure and Its Applications. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems (CHI EA '09)*. ACM, New York, NY, USA, 4369–4374. DOI: http://dx.doi.org/10.1145/1520340.1520668

13. Sven Mayer, Huy Viet Le, and Niels Henze. 2017. Estimating the Finger Orientation on Capacitive Touchscreens Using Convolutional Neural Networks. In *Interactive Surfaces and Spaces*. Brighton, UK. DOI: http://dx.doi.org/10.1145/3132272.3134130

14. Simon Rogers, John Williamson, Craig Stewart, and Roderick Murray-Smith. 2011. AnglePose: Robust, Precise Capacitive Touch Tracking via 3D Orientation Estimation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2575–2584. DOI: http://dx.doi.org/10.1145/1978942.1979318

15. Stuart Taylor, Cem Keskin, Otmar Hilliges, Shahram Izadi, and John Helmes. 2014. Type-hover-swipe in 96 Bytes: A Motion Sensing Mechanical Keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1695–1704. DOI: http://dx.doi.org/10.1145/2556288.2557030

16. Gerard Wilkinson, Ahmed Kharrufa, Jonathan Hook, Bradley Pursglove, Gavin Wood, Hendrik Haeuser, Nils Y. Hammerla, Steve Hodges, and Patrick Olivier. 2016. Expressy: Using a Wrist-worn Inertial Measurement Unit to Add Expressiveness to Touch-based Interactions. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 2832–2844. DOI: http://dx.doi.org/10.1145/2858036.2858223

17. Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. 2009. User-defined Gestures for Surface Computing. In *Proceedings of the SIGCHI*

Conference on Human Factors in Computing Systems (CHI '09). ACM, New York, NY, USA, 1083–1092. DOI: http://dx.doi.org/10.1145/1518701.1518866

18. Robert Xiao, Julia Schwarz, and Chris Harrison. 2015. Estimating 3D Finger Angle on Commodity Touchscreens. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces (ITS '15)*. ACM, New York, NY, USA, 47–50. DOI: http://dx.doi.org/10.1145/2817721.2817737
19. Haimo Zhang and Yang Li. 2014. GestKeyboard: Enabling Gesture-based Interaction on Ordinary Physical Keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1675–1684. DOI: http://dx.doi.org/10.1145/2556288.2557362
20. 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