Metamorphe: Augmenting Hotkey Usage with Actuated Keys

Gilles Bailly, Thomas Pietrzak, Jonathan Deber, Daniel J. Wigdor

Métamorphe: Augmenting Hotkey Usage with Actuated Keys

Gilles Bailly[1] **Thomas Pietrzak**[2] **Jonathan Deber**[3] **Daniel Wigdor**[3]

[1] Quality and Usability Lab, Telekom Innovation Laboratories, TU Berlin, Berlin, Germany
[2]Université Lille 1, Lille, France
[3]Department of Computer Science, University of Toronto, Toronto, Canada
gbailly@mpi-inf.mpg.de pietrzak@lifl.fr {jdeber, dwigdor}@dgp.toronto.edu

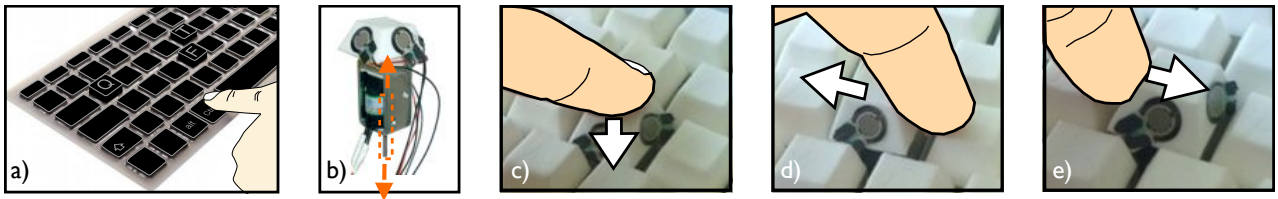

Figure 1. a) The Métamorphe keyboard raises a subset of hotkeys when users press the CTRL key. b) Each key can be individually raised with an embedded solenoid, and contains force sensors that allow a variety of novel gestures to be performed on the key. c-d-e) For example, the 'F' key can be pushed down to select "Find", or pushed left or right to select variations of that command.

ABSTRACT

Hotkeys are an efficient method of selecting commands on a keyboard. However, these shortcuts are often underused by users. We present Métamorphe, a novel keyboard with keys that can be individually raised and lowered to promote hotkeys usage. Métamorphe augments the output of traditional keyboards with haptic and visual feedback, and offers a novel design space for user input on raised keys (e.g., gestures such as squeezing or pushing the sides of a key). We detail the implementation of Métamorphe and discuss design factors. We also report two user studies. The first is a user-defined interface study that shows that the new input vocabulary is usable and useful, and provides insights into the mental models that users associate with raised keys. The second user study shows improved eyes-free selection performance for raised keys as well as the surrounding unraised keys.

Author Keywords

Augmented keyboard; height-changing keys; hotkeys; shape-changing interfaces; user-defined gestures.

ACM Classification Keywords

H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces – Interaction Styles.

INTRODUCTION

Hotkeys are an efficient way to select commands with a keyboard. In comparison to mouse-based techniques such as pull-down menus or toolbars, hotkeys offer several advantages: they are fast, particularly for repetitive actions (e.g., multiple "Copy"/"Paste" operations [18,19,26]); they let users interact "in place", minimizing mouse movements between graphical widgets [8]; and they do not consume valuable screen real estate [18]. Additionally, hotkeys can reduce hand movements between the keyboard and the mouse [26], increasing the speed of text entry and reducing the risk of musculoskeletal inflammation of the hand [14].

Despite these advantages, many users – even experienced ones – do not transition from the use of a GUI to the use of hotkeys [19]. This failure to transition has been attributed to an anticipated temporary "performance dip" (also known as the "gulf of incompetence") that users may not wish to attempt to bridge [37,41]. In addition, some users are not aware of hotkeys, while others find them a challenge to learn due to non-intuitive mappings [9,19]. Mapping issues may be due, in part, to the fact that any given key could be considered suitable for more than one command, forcing some commands to move to less intuitive keys [41]. Such challenges mean that many users choose traditional menus over hotkeys, ultimately preferring to maintain short-term productivity over long-term efficiency [6].

To overcome some of these challenges, we designed and built Métamorphe (see Figure 1), a novel shape-changing keyboard. While Métamorphe retains the form factor, layout, and tactile qualities of a traditional keyboard, it embeds solenoids that can raise individual keys, and adds force sensors to the sides of the keys, allowing a variety of novel gestures to be performed, increasing expressiveness.

These novel capabilities also provide new input and output channels that favor the guessability, learnability, and eyes-free selection of hotkeys.

The novel input and output capabilities raise interesting research questions: Is the increased interaction vocabulary useful and usable? How should designers assign commands to the new input primitives? Do raised keys improve performance for eyes-free key selection? To answer these questions we conducted two user studies. The first was a user-defined interface study that 1) demonstrated that users readily incorporate the novel input capabilities of raised keys, 2) provided insight into the mental models associated with their use, and 3) provided a set of user-defined hotkeys for Métamorphe. The second study evaluated the impact of raised keys on the eyes-free selection of hotkeys, and showed that 1) raised keys themselves are easier to locate on a keyboard than unraised keys, and 2) raised keys act as haptic landmarks, making it easier to locate neighboring, unraised keys. Finally, we present future directions for Métamorphe with potential scenarios.

RELATED WORK

In creating Métamorphe, we sought to build on related work in the areas of hotkey learning, enhanced keyboards, and shape-changing interfaces.

Hotkey Learning

While numerous studies suggest the importance of hotkeys as a keyboard tool [18,26], only a few solutions have focused on supporting hotkey learning [9,17,28]. Grossman *et al*. introduced audio feedback after the selection of a menu item to expose users to hotkeys [9]. They also proposed cost-based approaches, which disable the menu in order to force users to use hotkeys. While efficient [37], this strategy can frustrate users. HotkeyCoach combined feedback with a cost-based approach [17]: after every mouse-activated command, a pop-up window appeared illustrating the corresponding hotkey. Users could not continue until they used the hotkey or closed the pop-up.

Métamorphe aims to encourage hotkey usage by expanding the space of possible mappings between key and function, thus providing a richer input vocabulary to promote both the guessability and selection of hotkeys.

Enhanced Keyboards

Over the past 30 years, keyboards have remained essentially the same, despite increases in the variety and complexity of software [2,24]. Research on keyboards has investigated ergonomic designs, enhanced layouts, and new capabilities [21,24]. In this section, we review two areas of keyboard work: keyboards with novel input capabilities, and those with novel output capabilities.

Keyboard Input Capabilities

Dedicated function keys. The Xerox Star keyboard [3] and Microsoft's Office Keyboard [24] improve hotkey selection by providing keys dedicated to particular functions. Through haptic feedback, users are able to locate and select these keys eyes-free. While useful, maintaining a 1:1 mapping between functions and dedicated keys is realistic only for a small number of functions. In contrast, Métamorphe uses existing keys, supports both system-wide and application-specific hotkeys, and provides additional expressiveness by enabling different gestures on keys.

Finger-sensing and pressure input. Finger-sensing capabilities allow keys to not only be *pushed* and *released*, but also *touched*. This additional input state can be used to preview information, manipulate virtual objects, or perform gestures [4,15,33,40,42]. Force sensors further extend finger-sensing capabilities by offering a continuum of states between *touched* and *pushed*. Keyboards with force sensors have been used in various contexts such as gaming [7], user authentication [22], and emotional instant messaging [7]. Similarly, Métamorphe augments existing keys with force sensors, but places them on the sides of the keys to allow new gesture capabilities and new interactions.

Other additions to the keyboard have generally served specialized roles. For instance, the IBM TrackPoint [35] is a small rubber nub to the center of the keyboard, which is used as an isometric joystick to control the cursor in the absence of the mouse. Some extensions have been proposed using a key [35] or providing haptic feedback [5].

Keyboard Output Capabilities

*Visual Keyboard*s. The Optimus [27], Microsoft Adaptive Keyboard [25], and TDK [4] keyboards contain small screens on each key that can display application-specific icons or information. While these visual enhancements encourage the recognition of hotkeys, they also force users to divide their attention between the screen and the keyboard, which can be tiring and time-consuming [24].

Haptic feedback. Active haptic feedback is often used to increase the accuracy of virtual keyboards [12] or non-physical buttons [23] on touch surfaces. In the context of physical keyboards, force feedback has been proposed as a means to improve a user's comfort [36] and to prevent errors during text entry [11]. Métamorphe builds on this work by dynamically changing the height of raised keys.

Shape Changing

"A shape-changing interface uses physical change of shape as input or output" [31]. Shape-changing versions of mice [16], displays [1,29,34], keyboards [32], and interactive surfaces [10,38,39] have been proposed. Lumen [29] is a low-resolution interactive display where each pixel can move up and down. The Reflex Keyboard [32] changes shape, but does so by altering the relative positions of the two halves of the keyboard, with the goal of diminishing wrist pain. The Métamorphe's shape-changing keyboard differs from these earlier offerings, as its rising keys and force sensors provide a unique modality of shape changes.

In summary, Métamorphe offers an enhanced keyboard to promote hotkey usage. It differs from existing keyboards by providing a novel, height-changing mechanism that provides haptic feedback and enables new key gestures.

THE MÉTAMORPHE PROTOTYPE

The Métamorphe prototype is shown in Figure 2-bottom. It can individually raise any subset of keys (e.g., common hotkeys when a user presses the Control key). In this section we describe this height-changing mechanism, its integration into a traditional keyboard, and the various human factors that informed its design.

Métamorphe keys are raised and lowered using BLP Series 134 6V (30Ω) push-type solenoids mounted under each key (Figure 1-b). While other technologies, such as linear motors, shape memory alloys, or piezo ceramics can also provide linear motion, solenoids were selected because they are inexpensive, fast, stiff, small enough to fit under each key, and provide sufficient motion. However, solenoids have two limitations: they make a noise (roughly the same volume as a keystroke), and require continuous power to maintain a key in the raised position, necessitating heat dissipation. Despite these limitations, solenoids were ideal for the creation of our prototype, but further research and development would be required for a shelf-ready commercial implementation.

Each Métamorphe key has four functional positions. One dimension is set by the computer: *raised* (the key is higher than the rest of the keys) or *unraised* (the key is in its regular position). The other dimension (*pushed* or *not pushed*) is controlled by the user's input. We equipped our prototype with eight functioning dynamic keys, which is sufficient to allow a thorough exploration of the Métamorphe design space, and to conduct user studies.

In addition to *pushing*, dynamic Métamorphe keys allow users to perform a large inventory of key gestures, as shown in Figure 2-top. Gestures are detected by force sensors attached to the four side faces of the keys (see Figure 2).

Keyboard Integration

The Métamorphe prototype is built atop a regular HP KU-0316 USB keyboard. It reuses both the mechanical (i.e., the collection of switches) and the electronic (i.e., the circuitry that converts the output of those switches to the appropriate USB signals) components of the original keyboard.

Switches. The existing keyboard uses a dome switch to detect the *pressing* of each key. A dome switch contains a 4 mm dome that acts as a spring to push the key back to the default position when a user removes their finger. Our solenoids are positioned on top of these springs, allowing the keyboard's existing components to detect key presses in both the raised and unraised states.

Keyboard frame. We designed and 3D-printed a custom keyboard frame to vertically guide the solenoids. The actuated keys can be placed in any position on the keyboard. We also 3D-printed non-dynamic keys that fit into the frame to fill in the remaining spaces.

Electronics. Because Métamorphe reuses the existing keyboard's circuitry, it appears to its host computer as a regular USB keyboard and is compatible with all applications. An Arduino Nano, connected via a second USB cable, controls the solenoids and relays force sensor data.

Figure 2. Top: the physical manipulations that can be detected on each key. Bottom: our Métamorphe prototype.

Firmware and API. We developed a communication protocol between the keyboard and the computer, and wrote a C++ library for Linux, OS X, and Windows that encapsulates the communication and handles the mapping between the key IDs and the solenoids.

Human Factors

Actuation force. We conducted a series of tests to determine the ideal voltage to power the solenoids. They require sufficient mechanical power to lift the key cap, their own weight, and the sensors. 6V (suggested by the solenoid's specifications) is sufficient to hold the key in the raised position, but it is not sufficient to lift it. Thus, we briefly power the solenoids with a higher voltage (19V for 50ms) to raise the key, and then switch to 5V to maintain its position. We found that this provided sufficient force to raise the key, without fear of damaging the solenoids.

Which keys. Availability of parts limited the number of actuated keys in the prototype. This led us to consider what might otherwise have escaped notice: it may make sense to restrict the number of raised keys. Certainly, if all keys rise simultaneously, the value of the actuation is nil. Our two studies provide guidance in selecting keys for actuation.

How far to rise. The magnitude of the height change for actuated keys posed a critical design factor for Métamorphe. A key that is not raised enough may be overlooked, while a key that rises too much may impede ease of use, or interfere with unraised keys on the keyboard. We conducted a pilot study with 12 participants to evaluate key heights. We found that 1) at a height of less than 3 mm, the difference in height becomes difficult to perceive haptically, and that the comfort of the side push gestures decreases, and 2) the comfort of a *Top Push* decreases as the height increases to more than 4 or 5 mm. Based on these findings, we concluded that an optimal compromise would be a key height of 4 mm.

Retaining the properties of a keyboard. Métamorphe retains a similar form factor, layout, and tactile quality to a traditional keyboard. It functions normally as a text-entry device, and its spring threshold allows users to rest their fingers on the keys without triggering input [21,44]. When all the keys are in the lowered state, Métamorphe is virtually indistinguishable from a regular keyboard.

Passive-haptic feedback. Keyboards inherently provide some haptic feedback because of their physical keys. Métamorphe expands the range of possible haptic feedback. While it would be possible to use the raising of a key as a source of active haptic feedback (e.g., moving a key under the user's finger to attract their attention), we instead focus on using the raised/unraised state as a source of passive haptic feedback for haptic exploration (e.g., allowing a user to feel which keys are available as hotkeys by their state).

MÉTAMORPHE AND HOTKEYS

Métamorphe promotes hotkey selection in two fundamental ways. First, it applies lessons from prior work to make hotkeys easier to learn. Second, it increases expressiveness by adding raised keys, enabling new gestures that allow more hotkey actions to be defined.

Hotkey Finding

Métamorphe assists novices in identifying favorite or frequent hotkeys through both haptic and visual feedback.

Haptic Feedback

Haptic feedback has been shown to be remarkably fast and accurate as a tool to locate and recognize objects by touch alone [20]. On all keyboards, users can use their spatial memory and proprioception to develop a rough idea of the physical location of keys. However, Métamorphe's haptic feedback is able to provide additional assistance to accurately relocalize hotkeys.

Huynh *et al*. identify three haptic search methods: *scan*, *direct*, and *landmark* [13]. The *direct* and *landmark* search strategies are relevant for the relocalization of hotkeys.

The *direct* strategy uses a tactile or kinesthetic cue for error correction and confirmation of the target location [13]. When a key is raised, the height difference compared to other keys provides haptic feedback and confirms to users that the correct key has been reached.

The *landmark* strategy uses a raised key as an eyes-free landmark to help locate a nearby unraised target key. Users reach the raised haptic landmark, and then perform a quick ballistic movement towards the target location [13].

Visual feedback

In addition to its haptic feedback, a raised key also provides visual feedback, since it looks distinct from the surrounding keys. Although expert users generally prefer to focus on the screen and select hotkeys eyes-free, visual cues can be useful, particularly for novice users [4,27].

Expanded Hotkey Vocabulary

In addition to haptic and visual feedback, raised keys also introduce a novel design space for hotkey input.

Pressing and Squeezing of Raised Keys

On a standard keyboard, a key can be *pushed*, *held*, and *released*. However, Métamorphe's raised keys allow users to interact with several faces of the raised keys, enabling the 9 new gestures shown in Figure 3. Six of these new gestures have been implemented in our current prototype (Figure 2).

Users can *directionally push* (or *side push*) any of the four sides of a key, as illustrated in Figure 3-c-f. Directional pushes require only one finger, and allow for varying placement of the finger on the side of the raised key. Users can also *squeeze* a key with two fingers along the X-axis (a *Left-Right Squeeze* (*LR*), Figure 3-g) or the Y-axis (a *Front-Back Squeeze* (*FB*), Figure 3-h). Raised keys can also be rotated *Clockwise* (*CW*) and *Counterclockwise* (*CCW*) (Figure 3-i-j). Finally, users can *Pull* a raised key a small distance upwards (Figure 3-b). This novel vocabulary of gestures dramatically increases the expressiveness of the keyboard's input.

Number of Hotkeys

The number of single-key hotkeys is constrained by the number of keys available on a keyboard. This pool can be increased with multi-key hotkeys (e.g., using not only Control and another key, but also Control-shift and another key). However, multi-key hotkeys require additional finger coordination [24]. By allowing four new single-finger side push gestures per key, Métamorphe increases the number of available single-finger hotkeys. Two finger gestures (e.g., the squeezes) further expand the number of available hotkeys without requiring additional modifier keys.

Discrete & Continuous Control

Each gesture on our raised keys can be performed with different levels of force, offering two potential advantages. First, setting a force threshold allows the system to ignore false-positive activations when the user touches a key but does not mean to select it. Second, force sensors allow a user to select a command and *continuously control* a parameter in the same gesture. For instance, a *FB squeeze* of the 'Z' key could select a zoom command, while the level of force applied could control the rate of zooming. Other parameters (e.g., velocity or duration) could also provide additional customization of the command.

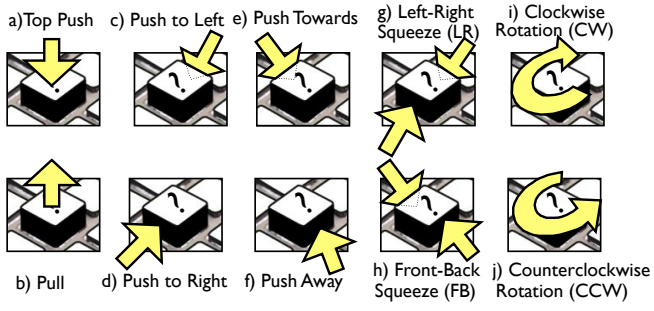

Figure 3. Design space of raised key gestures.

USER STUDY 1: USER-DEFINED HOTKEYS

Increasing the input vocabulary offers increased flexibility to map commands to hotkeys. For example, semantically-related commands could be assigned to a single raised key (e.g., "Save" and "Save As" could be assigned to different gestures on the same key 'S'). Symmetric gestures (e.g., opposing pushes on the same key) could also be assigned to dichotomous commands (e.g., "Previous" and "Next") [43].

We conducted a study to 1) determine how users would make use of the increased vocabulary afforded by the raised keys, 2) investigate the mental models associated with them, and 3) provide guidance to designers about how commands could be assigned to them. We adopted a user-defined interface approach, which involves presenting "the *effects* of actions to participants and elicits the *causes* meant to invoke them" [43]. Simply put, users are shown a video of the results of a command, and are then asked to indicate the input (i.e., the key and gesture) that they thought should elicit that result. This approach can be helpful in identifying users' underlying mental models since participants suggest their own commands, metaphors, and logical groupings.

We followed a similar methodology to Wobbrock *et al*. [43], but made one important modification. The existing methodology did not inform participants of the entire set of commands at the start of the study, causing participants to attempt to locally optimize individual mappings without considering the other commands they had yet to encounter. However, we believed that participants would likely choose to group semantically similar commands to the same gesture or key, if they were aware of all of the commands. Thus, we modified the procedure from the previous work such that: 1) participants were shown every command *before* they began defining their hotkeys; and 2) participants were given a worksheet where they recorded their desired keys and gestures, and could modify their choices as new information became available. This may come at the cost of reducing one form of "guessability" of our user-defined hotkey set, since we no longer report users' first, independent guesses. However, we believe that this will trade-off with an accelerated learning of subsequent commands (e.g., if the hotkey to "Align Left" is a *Push to Left* on the 'A' key, users will likely propose that "Align Right" is a *Push to Right* on the same key).

Participants and Apparatus

Twenty participants (11 females), ages 19 to 70 (mean=29, sd=12), were recruited via posters and word-of-mouth, and paid $20 for participation. All were infrequent hotkey users, self-reporting the use of an average of 4.8 (sd=2) hotkeys.

Stimulus videos were presented using a 20" monitor and standalone speakers. Following the protocol of Wobbrock *et al*. [43], we used a mockup, rather than the actual prototype, to avoid creating participant bias due to any particular design elements of the Métamorphe prototype. The mockup consisted of a sheet of paper printed with an American laptop-style keyboard layout and a transparent 16x16x4 mm plastic square that could be placed anywhere on the keyboard to simulate a raised key.

Procedure

An experimenter briefed each participant, and demonstrated a set of 10 gestures that can be performed on raised keys (shown in Figure 3). The experimenter explained the concept of *thinking-aloud*, and asked the participant to do so during the experiment. He then presented the participant with the 42 stimulus videos to familiarize the participant with the complete set of commands.

Our set of 42 commands (shown in Table 2) consisted of 25 commands used by Wobbrock *et al*. [43] (two commands specific to direct-manipulation were removed) and 17 new commands, organized in five semantic groups: *alignment* ("Align Left", "Align Right", "Align Top", "Align Bottom", "Align Middle", and "Align Justify"), *finding* ("Find", "Find Previous", and "Find Next"), *saving* ("Save", "Save As", and "Save All"), *audio* ("Play", "Pause", "Increase Volume", and "Decrease Volume"), and *clipboard* ("Copy", in addition to Wobbrock *et al*.'s existing "Cut" and "Paste"). The new commands were chosen to reflect traditional menu-based commands with semantically related groups and collisions (i.e., multiple commands beginning with the same letter).

Commands were then presented again, one-at-a-time. For each command, participants were asked to choose a key and a gesture they felt best *fit* with that command. They would then 1) demonstrate the gesture (placing the plastic square on the keyboard layout and performing the gesture); 2) fill in the worksheet with their chosen key and gesture; and 3) think-aloud their reasoning in order to provide the experimenter with an understanding of their mental model and selection strategies. If participants failed to think-aloud, they were prompted to do so by the experimenter. At any time, they were permitted to return to a previous command and modify their choice of key or gesture.

Design

Commands were presented to each participant in a randomized order (without replacement). The overall design of the experiment was: 20 participants x 42 commands = 840 hotkey selections.

Results

We first report quantitative findings from the study, followed by a qualitative discussion of the participants' mental models and selection strategies. Finally, we report a set of user-defined hotkeys.

Quantitative Results

Agreement. Wobbrock *et al*. define agreement as a score between 0 and 1 "that reflects, in a single number, the degree of consensus among participants" [43]. In our work, we distinguish three types of agreement for each command: 1) *key agreement* (participants selected the same key), 2) *gesture agreement* (selection of the same gesture), and 3) *combined agreement* (selection of the same key and gesture). Using Wobbrock *et al*.'s method, we calculated the three agreement scores for each command, which are summarized in Figure 4. We noted that highly directional commands (e.g., "Align Left") tended to have a high gesture agreement, while strongly mnemonic commands (e.g., "Copy") tended to have a high key agreement.

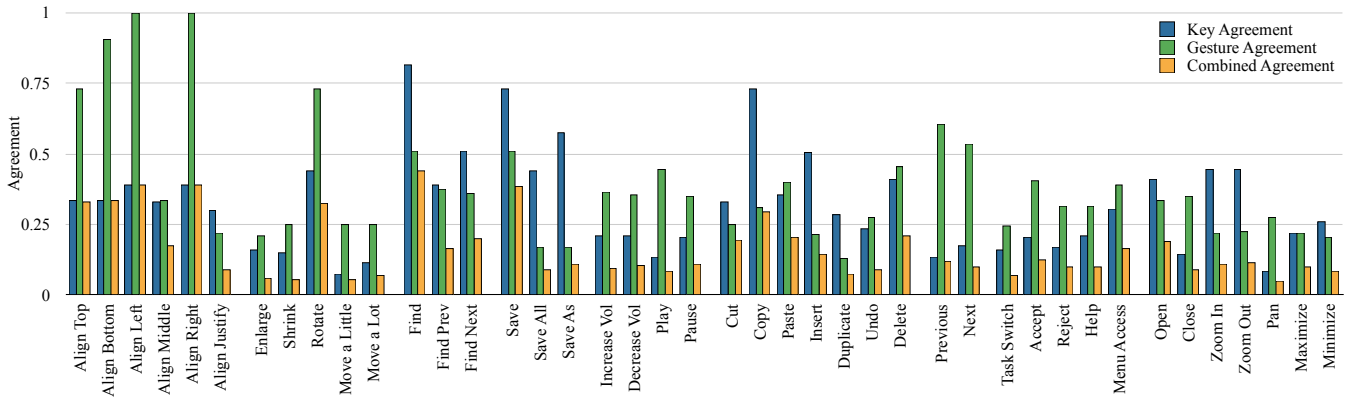

Figure 4. Key, gesture, and combined agreement scores for each command, sorted by semantic group.

Distribution of Keys and Gestures. Excluding two participants that explicitly tried to minimize the number of keys used (using 4 and 7 keys for the entire set of commands), the average number of keys used was 21.6 keys (sd=2.9), a mean of 1.94 commands per key. The average number of different gestures used was 9.3 (sd=1.2); 11 of 20 participants used all 10 gestures. However, some gestures were clearly preferred, as shown in Figure 5.

Grouped Commands. Our command set included two groups containing a primary command and two variants: the *finding* group ("Find", "Find Next", and "Find Previous"), and the *saving* group ("Save", "Save As", and "Save All"). In both cases, participants tended to assign the primary command (e.g., "Save") to *Top Push*, and the variants to different gestures on the same key. For example, 18 participants assigned at least two of the saving commands to the same key, with 14 of them assigning all three commands to the same key. 13 of the 18 that assigned multiple commands to one key picked *Top Push* for the primary command.

Directional Commands. For highly directional commands (e.g., "Pan" or "Move a Lot"/"Move a Little"), participants tended to interact with a raised key as a rate-controlled pointing device (several participants explicitly termed it a "joystick"), and mentioned that they would like to use it to move the view or object. For instance, 15 participants used a key as a joystick when panning. Similarly, 15 participants assigned multiple "Align" commands to a single key, pushing the key in the direction of the desired alignment. These results are consistent with previous findings that participants in user-defined gesture studies tend to gravitate towards physical manipulation metaphors [43].

Dichotomous Commands. Some commands can be classified as dichotomous pairs (e.g., "Shrink"/"Enlarge", "Previous"/"Next", *etc*.). Each pair of keys and pair of gestures associated with dichotomous commands can be divided into three categories: 1) *identical* (i.e., they use the same key or the same gesture), 2) *opposite* (e.g., '<' and '>', or *Push to Left/Right*), and 3) *different*. An ANOVA reveals a significant effect for the gesture*key interaction on the percentage of hotkeys used for dichotomous commands ($F_{4,76}$=52.6, p<0.0001). Pair-wise comparisons indicate that participants use opposite gestures on the same keys significantly more often than all other strategies to map dichotomous commands (Table 1). For such commands, participants preferred directional pushes (36 occurrences of *Push to Left/Right* and *Push Away/Towards*), although *CW/CCW* (24 occurrences) and *Top Push/Pull* (16 occurrences) were also used.

		Gesture		
		Identical	**Opposite**	**Different**
Key	**Identical**	4.0%	56.0%	6.0%
	Opposite	7.0%	8.0%	1.5%
	Different	3.0%	9.0%	5.5%

Table 1. Breakdown of key and gesture assignments for dichotomous commands.

Mental Models and Selection Strategies

Having reviewed quantitative results, we now examine strategies employed by participants in making their selections of keys and gestures, in order to gain insight into the mental models associated with the mappings.

For the majority of the 42 commands, most participants decided on an appropriate key first, and then chose an appropriate gesture, although some commands (e.g., "Rotate"), tended to elicit a gesture choice first.

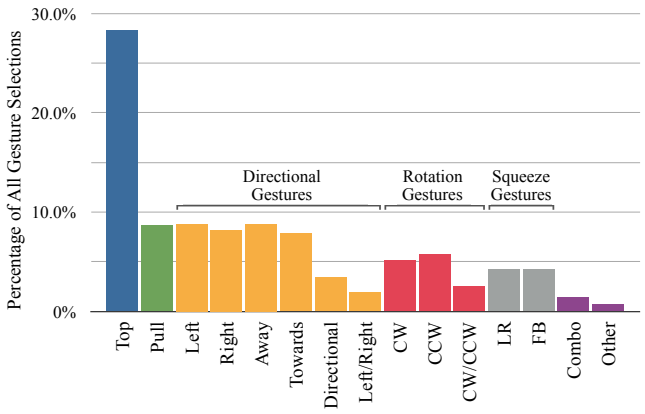

Figure 5. Distribution of gesture selections. Entries containing a slash (e.g., "Left/Right") indicate that the user chose to use either of the listed gestures for a command.

Strategies for Key Selection

An analysis of the think-aloud data revealed six primary strategies for key selection.

Mnemonics. This was the most common basis for choosing a key. Participants selected keys based on the command's first letter (e.g., 'S' for "Save"), significant letter (e.g., 'L' for "Align Left", or 'W' for "Task S**w**itch"), or synonym (e.g., 'D' for "Shrink" and "Enlarge" because both are forms of "dilation").

Prior Experience. Many selections were based on previous experience (e.g., 'V' for "Paste"), although a few participants used the opportunity to "correct mistakes" in existing hotkey assignments, or take better advantage of the raised keys (e.g., moving "Paste" to an opposing gesture on 'C').

Physical Location. Participants used the physical location of a key on the keyboard (e.g., F5 for "Menu Access" because the key is located at the top of the keyboard, mirroring the location of the menu bar).

Glyph Affordances. Participants considered the glyphs printed on a key (e.g., '%' for "Align Middle" because "it's like two circles with a line in the middle"). Participants often selected keys based on their secondary symbols, despite the fact that they did not use the shift key as part of their hotkey. However, some participants used gestures to disambiguate symbols on keys (e.g., P6 picked *Push Away* on '6' because he wanted to use the '^' symbol).

Physical Constraints. Some participants considered a key's physical constraints (e.g., the difficulty of rotating the spacebar) and some used placement strategies designed to reduce the risk of accidently hitting a "dangerous" key.

Semantic Grouping. Finally, semantic groupings were also common. Participants would make one key assignment based upon one of the above strategies, and then assign semantically related commands to the same key (e.g., the aforementioned assignment of "Paste" to 'C', or picking 'V' for "Increase Volume", followed by the subsequent assignment of "Play" to the same "music functions" key).

Strategies for Gesture Selection

Participants typically used *Top Push* as their default gesture, and opposing gestures for dichotomous commands (see Figure 4 and Table 1). We also identified three other gesture selection strategies.

Avoiding Errors. Some participants intentionally chose gestures they saw as difficult or ergonomically challenging (e.g., *Pull* or *LR*) to invoke commands deemed to have significant or irreversible actions (e.g., "Delete").

Directional Gestures. Participants used directional gestures for a variety of actions, but some expressed a reluctance to use directional gestures for non-directional actions (e.g., assigning "Copy" to a *Push to Left*).

Metaphors. Participants frequently used analogies and metaphors for selecting gestures, including physical gestures (e.g., *CW/CCW* to "Rotate") and metaphorical gestures (e.g., *Pull* to summon a help system based on the concept of pulling an emergency assist cord in a hospital, or *FB* to "squish" text). Some also chose gestures based on similarities to multitouch gestures they already knew (e.g., *LR* uses a finger motion that is similar to pinch-to-zoom).

Finally, some participants proposed novel combinations of gestures, extending the expressiveness of the initial gesture set. These included wiggling keys (i.e., rapidly moving the key left and right), combining a squeeze with a rotation to make the rotation "more powerful", and combining a *Top Push* with a directional movement to provide additional context for the operation (e.g., using a directional push to align a single object, but a *Top Push* combined with a directional push to "select all" and align every object at once). Other participants introduced concepts such as pressure (pushing harder implied a greater magnitude for an operation), duration (long pushes vs. short pulses), and repeated gestures (e.g., a pair of *Top Pushes* for "Duplicate"). We found that the open-ended nature of the user-defined gesture methodology with a paper prototype was an extremely productive tool to elicit these extensions.

Due to the finite size of our command set, we of course could not encompass all of the activities that users perform on a computer. Although we attempted to create a representative set of commands, there were some commands that many or most participants indicated that they would not normally use (e.g., "Save All"), and some commands that some felt were missing (e.g., "Print"). It is possible that the presence or absence of additional commands could impact the set of key assignments (e.g., the presence of "Print" might have impacted the use of 'P'). We leave this question to be answered in future work.

A Set of User-Defined Métamorphe Hotkeys

To determine the most common key and gesture selections for each command, we analyzed the data in two ways. First, we evaluated the key and gesture independently (i.e., for each command, pick the most common key, and then pick the most common gesture). In cases where multiple keys or gestures were selected the same number of times, we considered all possible permutations (e.g., for "Align Middle", both *FB* and *Top Push* were each selected 8 times; since 'A' was the most frequent key, we consider both A-*FB* and A-*Top*). The second analysis considered the key and gesture as a pair (i.e., for each command, select the most common tuple of key and gesture). The former approach more accurately captures the individual keys and gestures, but could result in hotkey assignments that were not actually chosen by any users, while the latter could result in a simple plurality of a small number of users overriding the general trends of the remaining pool.

We compared the resulting sets and developed the user-defined hotkey set shown in bold in Table 2. For 30 of 42 commands, both methods yielded a single identical hotkey. For eight commands, although one or both methods yielded multiple hotkeys with the same number of occurrences, there was only a single hotkey that was obtained by both methods, which was selected for our set.

For the remaining four commands, there was no clear agreement between the two methods. As such, while an HCI practitioner could certainly select an appropriate compromise as a hotkey, based on our dataset there does not appear to be a purely user-defined alternative that is dominant. In addition, it is interesting to note that there are some choices that appear actively suboptimal. For example, "Previous" and "Next" have opposite gestures, but are placed on different keys. More significantly, "Accept" and "Reject" have mixed mental models; "Accept" has been assigned to 'Y' (based on a yes/no mental model), while "Reject" has been assigned to 'R' (based on an accept/reject mental model); this hybrid mental model would never be chosen intentionally. Although these inconsistencies demonstrate some of the limitations of a strictly user-defined interface, the approach still provides invaluable insight into users' mental models.

In conclusion, this user study shows: 1) participants readily assign multiple commands to the same key, and 2) participants exploit the range of novel gestures introduced by a raised key, particularly to highlight semantic relationships (e.g., dichotomous commands). Additionally, it provides an example of a set of user-defined hotkeys for Métamorphe.

USER STUDY 2: EYES-FREE SELECTION

We also conducted a user study to compare the speed and accuracy of eyes-free hotkey selection on Métamorphe and a traditional keyboard. Specifically, our research questions were: 1) Do raised keys improve performance for eyes-free selection? 2) Do raised keys impact the performance of selecting surrounding unraised keys? 3) How easy is it to push the side of a raised key?

Procedure

On Métamorphe, users were asked to perform the following actions: *Top Push* on both a raised and unraised key, and the four side pushes on a raised key. On the regular keyboard without raised keys, we replaced the side pushes with additional *Top Pushes*. The stimulus for each trial was a visual depiction of a keyboard that highlighted the target key, as well as the locations of all raised keys. We used this visual depiction to simulate the case where a user is already familiar with the command-to-hotkey mapping, and has a rough idea of the location of the target on the keyboard. The 20 different target hotkeys used in the experiment are shown in Figure 6. There were two raised keys ('D' and 'O', yielding 4 x 2 = 8 possible side pushes) and 18 lowered keys distributed near the raised keys.

Participants received a 5 min. training phase, which allowed them to select each target once. They began each trial by pressing the spacebar with both hands. A stimulus appeared on-screen, and participants selected the indicated hotkey by pressing Control and the specified key. Participants were asked to select hotkeys eyes-free as quickly and accurately as possible. As soon as the top or side of a key was pressed, the trial was completed. They then moved their hands back to the spacebar, and began the next trial.

Participants & Apparatus

Twelve right-handed volunteers from our institution, ages 23 to 39 (mean=28.3, sd=5.2) participated in the experiment. We used a black box to mask Métamorphe from the participants to ensure eyes-free selection of keys.

Design

We used a within-subject design. Participants performed 4 blocks of 28 key selections for each keyboard. The order of keys within each block was randomized. The order of presentation of keyboards was counter-balanced between participants. The overall design was: 12 users x 2 keyboards x 28 targets x 4 blocks = 1344 selections.

Results

Keyboard. An ANOVA reveals a significant effect for *keyboard* on accuracy ($F_{1,11}$=19.01, p<0.001) A *post hoc* Tukey test shows that Métamorphe is significantly more accurate than the traditional keyboard (95% vs. 80%). An ANOVA also reveals a significant effect for *keyboard* on speed ($F_{1,11}$=54.14, p<0.0001). A *post hoc* Tukey test shows that Métamorphe is also significantly faster (2.6 s vs. 3.9 s).

Command	*Top Key(s) + Top Gesture(s)*	*Top Tuple(s) (if different)*	*Command*	*Top Key(s) + Top Gesture(s)*	*Top Tuple(s) (if different)*	*Command*	*Top Key(s) + Top Gesture(s)*	*Top Tuple(s) (if different)*
Align Top	**A-*Away***		Save	**S-*Top***		Previous	**P-*Left***	
Align Bottom	**A-*Towards***		Save All	**S-*Pull***	**S-*Pull***, S-*LR*	Next	**N-*Right***	
Align Left	**A-*Left***		Save As	**S-*Right***		Task Switch	tab-*Left*/*Right*	[4 tuples tied]
Align Middle	**A-*FB***, A-*Top*	**A-*FB***	Increase Vol	**V-*CW***		Accept	**Y-*Top***	
Align Right	**A-*Right***		Decrease Vol	**V-*CCW***		Reject	**R-*Top***	
Align Justify	J-*Pull*	J-*Top*, A-*Pull*	Play	**P-*Left***		Help	**H-*Top***	
Enlarge	**+-*Pull***, +-*CW*	**+-*Pull***, S-*CW*	Pause	**N-*Right***		Menu Access	**menu-*Top***	
Shrink	**S-*CCW***, --*CCW*	**S-*CCW***	Cut	**X-*Top***, C-*Top*	**X-*Top***	Open	**O-*Top***	
Rotate	**R-*CW*/*CCW***		Copy	**C-*Top***		Close	**C-*Top***	
Move a Little	M-directional	N-directional	Paste	**V-*Top***		Zoom In	**Z-*CW***	
Move a Lot	**M-directional**		Insert	**I-*Top***		Zoom Out	**Z-*CCW***	
Find	**F-*Top***		Duplicate	**D-*Top***		Pan	P-directional	[20 tuples tied]
Find Prev	**F-*Left***		Undo	**U-*Top***	**U-*Top***, Z-*Top*	Maximize	**M-*Pull***	
Find Next	**F-*Right***		Delete	**backspace-*Top***, D-*Top*	**backspace-*Top***	Minimize	**M-*Towards***, M-*CCW*, M-*Top*	**M-*Towards***, --*CCW*

Table 2. The most common hotkeys (using the *key + gesture* and *tuple* methods) for all 42 commands. Hotkeys in bold are our user-defined set. A blank entry in the *tuple* column indicates that this hotkey was the same as the *key + gesture* hotkey.

Distance from raised keys on Métamorphe. An ANOVA reveals a significant effect for *distance* (from the target to a raised key) on accuracy ($F_{2,33}$=7.78 p<0.001). A *post hoc* Tukey test shows that the accuracy at a distance of 0 keys (i.e., selecting a raised key) (98%), 1 key (95%), and 2 keys (97%) is significantly greater than 3 keys (82%). There is a significant effect for *distance* on time ($F_{3,33}$=58.56 p<0.0001). A *post hoc* Tukey test shows that time increases with the distance (d) from the raised key (d = 0: 1.7 s; d = 1: 2.4 s; d = 2: 2.8 s; d = 3: 3.6 s), indicating that users are indeed using the raised key as a landmark.

Side Pushes. An ANOVA reveals a significant effect for *gesture* on accuracy ($F_{4,44}$=7.19, p<0.0001). A *post hoc* Tukey test shows that *Top Push* is significantly more accurate (98%) than the other gestures (*Push to Right*: 88%; *Push to Left*: 74%; *Push Away*: 78%; *Push Towards*: 79%). An ANOVA also reveals a significant effect for *gesture* on time ($F_{4,44}$=10.03, p<0.0001), with a *post hoc* Tukey test showing that *Top Push* is significantly faster (1.8 s) than the other gestures (*Push to Right*: 2.4 s; *Push to Left*: 3.0 s; *Push Away*: 2.5 s; *Push Towards*: 2.1 s).

Observations.

Raised keys. While the quantitative results did not show differences between the different side pushes, participants reported that some gestures were more challenging to perform. 7 participants mentioned that *Push to Right* with the right hand or *Push to Left* with the left hand was slightly more difficult to perform because it requires an unnatural posture. However, one user also mentioned "it was easier to select a hotkey on the side of a key than a lowered key far from a raised key".

Unraised keys. All users first reached for the closest raised key before moving to the target unraised key, even if the use of this landmark required them to initially overshoot the target key. However, they adopted two different strategies of haptic exploration to perform the final movement from the raised key to the unraised key. The first strategy used a single finger, which was simply moved from one key to the other. The second strategy used two fingers. One finger (e.g., the ring finger) is used as a pivot on the raised key, while a second finger (e.g., the index finger) directly reaches for the unraised key. The distance between fingers is used as an indicator to approximate the distance between the raised key and the unraised key.

Figure 6. Heat map of selection speed (in s) for each target keys. The two raised keys are outlined. Insets show speeds for *Top Push* and the directional pushes on the raised keys.

Discussion

Performance. Our results show that Métamorphe improves eyes-free selection in both time (improvement of 33%) and accuracy (improvement of 18%) over a traditional keyboard. These results confirm that the additional passive haptic feedback provided by Métamorphe can help users to select hotkeys eyes-free.

Haptic exploration. Our observations confirm that users use the raised keys as haptic landmarks [13] to select unraised keys in proximity to a raised key. We also observed that users combine haptic exploration and proprioception to select hotkeys.

Neighborhood. The accuracy decreases significantly when the distance from the raised key is more than two keys; we therefore recommend placing raised key landmarks within that distance of any desired unraised key targets.

Sides of Keys. Finally, our results confirm that *Top Push* is faster and more accurate than the side pushes. Additionally, they do not reveal any performance differences among the side pushes. However, *Push to Right* with the right hand or *Push to Left* with the left hand were perceived as slightly more difficult.

FUTURE WORK

We envision several future applications for Métamorphe.

Discoverability of hotkeys. Métamorphe could enhance hotkey learning by raising applicable hotkeys (e.g., when the cursor hovers over an object). This could also help visually impaired users to explore available hotkeys.

Text input. Métamorphe could be adapted to predict the next character in a string of text and automatically raise that key to (subtly) prompt the user and avoid spelling errors. This capability would be most appropriate with short and constrained text entry (e.g., city names). In addition, rising keys could provide tactile hints when learning typing skills.

Security. Métamorphe could provide additional security features for password entry, since passwords could consist of both characters and gestures. This would increase the keyspace of inputs to the system, increase the difficulty of shoulder surfing, and could make passwords easier to remember because users could incorporate spatial memory.

Gaming. Raised keys could indicate a ready to use action (e.g., a spell), and the increased vocabulary could provide additional game controls.

Tangible Interfaces. Physical controls could be provided on-demand for a wide variety of tasks. For instance, an on-screen dialog with three slider widgets could raise three keys to provide tangible controls for each widget.

CONCLUSION

We have presented Métamorphe, a novel keyboard with height-changing keys that provides haptic and visual feedback. As an input device, Métamorphe also offers a novel vocabulary of gestures, which increases the number of available hotkeys and their expressiveness. To validate Métamorphe, we conducted two user studies. The first

study 1) showed that users chose to exploit the novel capabilities of the keyboard when assigning hotkeys, 2) provided us with insight into users' mental models, and 3) produced a set of user-defined hotkeys. A second user study 4) found that Métamorphe's raised keys are easier to select eyes-free than unraised keys, and 5) raised keys positively impact the ease of selection of nearby unraised keys. Finally, we defined promising areas for development for shape-changing keyboards.

ACKNOWLEDGEMENTS

This work was funded in part by the Alexander von Humboldt Foundation. We thank M. Amberg, J. Müller, M. Lorenzo, V. Meunier, B. Legouis, J. Hancock, I. Ben Said, and K. Wright.

REFERENCES

1. Bau, O., Petrevski, U., and Mackay, W. (2009) BubbleWrap: A Textile-Based Electromagnetic Haptic Display. *CHI EA'09*, 3607-3612.
2. Beaudouin-Lafon, M. (2004) Designing Interaction, not Interfaces. *AVI '04*, 15-22.
3. Bewley, W. L., Roberts, T. L., Sehroit, D., and Verplank, W. L. (1983) Human Factors Testing in the Design of Xerox's 8010 "Star" Office Workstation. *CHI '83*, 72-77.
4. Block, F., Gellersen, G., and Villar, N. (2010) Touch-Display Keyboards: Transforming Keyboards into Interactive Surfaces. *CHI '10*, 1145-1154.
5. Campbell, C. S., Zhai, S., May, K. W., and Maglio, P. P. (1999) What You Feel Must be What You See: Adding Tactile Feedback to the Trackpoint. *INTERACT '99*, 383-390.
6. Carroll, J. M., and Rosson, M. B. *Paradox of the Active User*. MIT Press, Cambridge, MA, USA, 1987, pp 80–111.
7. Dietz, P. H., Eidelson, B., Westhues, J., and Bathiche, S. (2009) A Practical Pressure Sensitive Computer Keyboard. *UIST '09*, 55-58.
8. Fitzmaurice, G., Matejka, J., Khan A., Glueck M., and Kurtenbach, G. (2008) PieCursor: Merging Pointing and Command Selection for Rapid In-place Tool Switching. *CHI '08*, 1361-1370.
9. Grossman, T., Dragicevic, P., and Balakrishnan, R. (2007) Strategies for Accelerating On-line Learning of Hotkeys. *CHI '07*, 1591-1600.
10. Harrison, C. and Hudson, S. E. (2009) Providing Dynamically Changeable Physical Buttons on a Visual Display. *CHI '09*, 299-308.
11. Hoffmann, D., Spelmezan, D., and Borchers, J. (2009) TypeRight: a Keyboard with Tactile Error Prevention. *CHI '09*, 2265-2268.
12. Hoggan, E., Brewster, S. A., and Johnston, J. (2008) Investigating the Effectiveness of Tactile Feedback for Mobile Touchscreens. *CHI '08*, 1573-1582.
13. Huynh, K., Stepp, C.E., White, L.W., Colgate, J.E., and Matsuoka, Y. Finding a feature on a 3D object through single-digit haptic exploration. *IEEE Haptics Symposium '10*, 83-89.
14. Jorgensen, A., Garde, A., Laursen, B., and Jensen, B. Using Mouse and Keyboard Under Time Pressure: Preference, Strategies and Learning. *Behavior and Information Tech. 21*, 5 (2002), 317–319.
15. Kato, J., Sakamoto, D., and Igarashi, T. (2010) Surfboard: Keyboard with Microphone as a Low-cost Interactive Surface. *UIST '10*, 387-388.
16. Kim, S., Kim, H., Lee, B., Nam, T.-J., and Lee, W. (2008) Inflatable Mouse: Volume-adjustable Mouse with Air- pressure-sensitive Input and Haptic Feedback. *CHI '08*, 211-214.
17. Krisler, B., and Alterman, R. (2008) Training Towards Mastery: Overcoming the Active User Paradox. *NordiCHI '08*, 239–248.
18. Kurtenbach, G. The Design and Evaluation of Marking Menus. *Ph.D. Thesis, Dept. of Computer Science, University of Toronto*, 1993.
19. Lane, D. M., Napier, A. H., Peres, C. S., and Sándor, A. The Hidden Costs of Graphical User Interfaces: The Failure to Make the Transition from Menus and Icon Tool Bars to Keyboard Shortcuts. *International Journal of Human-Computer Interaction 18*, 2 (2005), 133–144.
20. Lederman, S.J., and Klatzky, R. L. Hand Movements: A Window Into Haptic Object Recognition. *Cog. Psych. 19*, 3 (1987), 342-368.
21. Lewis, J. R., Potosnak, K. M., and Magyar, R. L. Keys and Keyboards. In M. Helander, T. K. Landauer, P. V. Prabhu. *Handbook of Human-Computer Interaction*. Elsevier, 1997, 1285-1311.
22. Loy, C., Lai, W., and Lim, C. Development of a Pressure-based Typing Biometrics User Authentication System. Tech. rep., University of Science Malaysia & MIMOS Berhad, 2005.
23. Lylykangas, J., Surakka, V., Salminen, K., Raisamo, J., Laitinen, P., Rönning, K., and Raisamo, R. (2011) Designing Tactile Feedback for Piezo Buttons. *CHI '11*, 3281-3284.
24. McLoone, H., Hinckley, K., and Cutrell, E. (2003) Bimanual Interaction on the Microsoft Office Keyboard. *INTERACT '03*, 49-56.
25. Microsoft Adaptive Keyboard. http://www.microsoft.com/appliedsciences/content/projects/uist.aspx
26. Nielsen, J. *Usability Engineering*. Morgan Kaufman, 1993.
27. Optimus Maximus Keyboard, ArtLebedev Studios. http://www.artlebedev.com/everything/optimus/
28. Peres, S. C., Ii, F. P. T., Fleetwood, M. D., Chung, P., & Paige-Smith, D. L. Keyboard Shortcut Usage: The Roles of Social Factors and Computer Experience. In *Proc. of Human Factors and Ergonomics Society 48th Annual Meeting 2004*, 803–807.
29. Poupyrev, I., Nashida, T., Maruyama, S., Rekimoto, J., and Yamaji, Y. (2004) Lumen: Interactive Visual and Shape Display for Calm Computing. *SIGGRAPH '04*, 17.
30. Poupyrev, I., Nashida, T., and Okabe, M. (2007) Actuation and Tangible User Interfaces: The Vaucanson Duck, Robots, and Shape Displays. *TEI '07*, 206-212.
31. Rasmussen, M. K., Pedersen E. W., Petersen, M. G., and Hornbæk, K. (2012) Shape-Changing Interfaces: A Review of the Design Space and Open Research Questions. *CHI '12*, 735-744.
32. Reflex Keyboard, Smartfish Technologies.
33. Rekimoto, J., Ishizawa, T., Schwesig, C., and Oba, H. (2003) PreSense: Interaction Techniques for Finger Sensing Input Devices. *UIST '03*, 203-212.
34. Richter H., and Wiethoff, A. (2011) EdgeMatrix: Remote Tactile Feedback on Interactive Surfaces Using a Shape Display. *TEI '11*.
35. Rutledge, J. D., and Selker, T. (1990) Force-to-Motion Functions for Pointing. *INTERACT '90*, 701-706.
36. Savioz, G., Markovic, M., & Pierrard, Y. (2011) Towards multi-finger haptic devices: A computer keyboard with adjustable force feedback. *ICEMS '11*.
37. Scarr, J., Cockburn, A., Gutwin, C., and Quinn, P. (2011) Dips and Ceilings: Understanding and Supporting Transitions to Expertise in User Interfaces. *CHI '11*, 2741-2750.
38. Weiss, M., Schwarz, F., Jakubowski, S., and Borchers, J. (2010) Madgets: Actuating Widgets on Interactive Tabletops. *UIST '10*, 293-302.
39. Weiss, M., Remy, C., and Borchers, J. (2011) Rendering Physical Effects in Tabletop Controls. *CHI '11*, 3009-3012.
40. Westerman, W. *Hand Tracking, Finger Identification, And Chordic Mani-pulation On A Multi-Touch Surface*. Ph.D Thesis, U. of Delaware, 1999.
41. Wigdor, D. and Wixon, D. *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*. Morgan Kaufman, 2011.
42. Wilson, A. D. (2006) Robust Computer Vision-Based Detection of Pinching for One and Two-Handed Gesture Input. *UIST '06*, 255-258.
43. Wobbrock, J. O., Morris, M. R., and Wilson, A. D. (2009) User-Defined Gestures for Surface Computing. *CHI '09*, 1083-1092.