

Palmrest+: Expanding Laptop Input Space with Shear Force on Palm-Resting Area

Jisu Yim
HCI Lab, KAIST
Daejeon, Republic of Korea
yimjisu99@kaist.ac.kr

Seoyeon Bae
HCI Lab, KAIST
Daejeon, Republic of Korea
s2yeon517@kaist.ac.kr

Taejun Kim
HCI Lab, KAIST
Daejeon, Republic of Korea
taejun.kim@kaist.ac.kr

Sunbum Kim
HCI Lab, KAIST
Daejeon, Republic of Korea
ksb4587@kaist.ac.kr

Geehyuk Lee
HCI Lab, KAIST
Daejeon, Republic of Korea
geehyuk@kaist.ac.kr

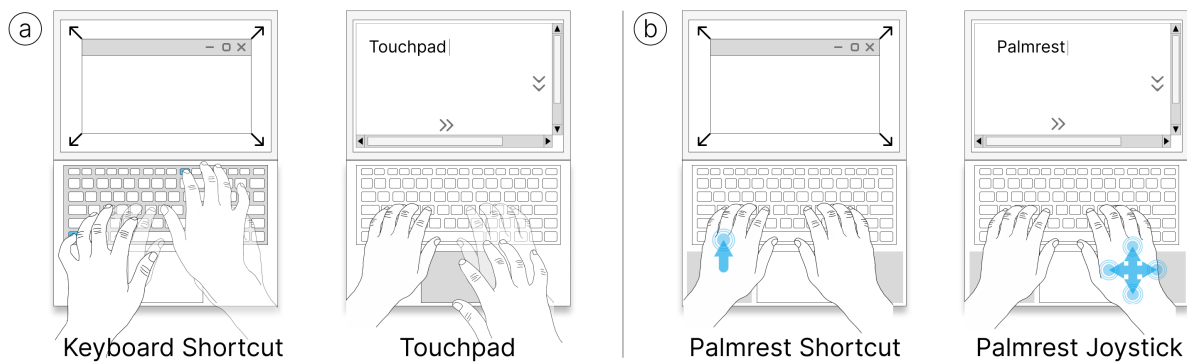


Figure 1: (a) Using a keyboard shortcut or touchpad in the middle of typing requires physical movement away from the home typing position (b) Palmrest+ interaction allows for subtle input while keeping the palms in the typing position by leveraging shear force applied by the palms: Palmrest Shortcut enables rapid command execution by subtly applying shear force, and Palmrest Joystick enables continuous value adjustment.

ABSTRACT

The palmrest area of laptops has the potential as an additional input space, considering its consistent palm contact during keyboard interaction. We propose Palmrest+, leveraging shear force exerted on the palmrest area. We suggest two input techniques: Palmrest Shortcut, for instant shortcut execution, and Palmrest Joystick, for continuous value input. These allow seamless and subtle input amidst keyboard typing. Evaluation of Palmrest Shortcut against conventional keyboard shortcuts revealed faster performance for applying shear force in unimanual and bimanual-manner with a significant reduction in gaze shifting. Additionally, the assessment of Palmrest Joystick against the laptop touchpad demonstrated comparable performance in selecting one- and two- dimensional targets with low-precision pointing, i.e., for short distances and

large target sizes. The maximal hand displacement significantly decreased for both Palmrest Shortcut and Palmrest Joystick compared to conventional methods. These findings verify the feasibility and effectiveness of leveraging the palmrest area as an additional input space on laptops, offering promising enhanced typing-related user interaction experiences.

CCS CONCEPTS

• Hardware → Sensors and actuators; • Human-centered computing → Gestural input.

KEYWORDS

input techniques, gesture, laptop computers, prototyping

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
UIST '24, October 13–16, 2024, Pittsburgh, PA, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0628-8/24/10
<https://doi.org/10.1145/3654777.3676371>

ACM Reference Format:

Jisu Yim, Seoyeon Bae, Taejun Kim, Sunbum Kim, and Geehyuk Lee. 2024. Palmrest+: Expanding Laptop Input Space with Shear Force on Palm-Resting Area. In *The 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)*, October 13–16, 2024, Pittsburgh, PA, USA. ACM, New York, NY, USA, 14 pages. <https://doi.org/10.1145/3654777.3676371>

1 INTRODUCTION

Laptops have become indispensable tools offering high-performance computing in a compact, lightweight, and portable form factor [22, 23]. For the decades of their widespread use, the standard design has remained unchanged, with reliance on the touchpad and keyboard for user input. In the meantime, the palmrest areas, located on each side of the touchpad, are typically in contact with the users' palms during keyboard typing for stability and comfort [29]. This offers a promising opportunity for supplementary input space, potentially facilitating seamless and subtle input while maintaining the hands in their home typing position.

When users need to perform a non-text input in the midst of keyboard typing, such as executing a certain command or adjusting a continuous value widget, they typically turn to the touchpad. However, using the touchpad mid-typing involves the physical cost of switching between the keyboard and touchpad, disrupting workflow and reducing efficiency [16]. While keyboard shortcuts may provide an alternative with lower switching costs, the excessive number of existing shortcuts often leads to unintuitive command-shortcut mappings, making many of them hard for users to recall [9, 17].

In this work, we propose Palmrest+, an input space for seamless and rapid input on a laptop in the middle of text entry. To minimize the physical movement during text entry, our approach leverages the shear force exerted on the palmrest area of the laptop, inspired by previous studies exploring shear force on surfaces [12, 13, 15, 30] for expanding the input space with subtle movements. We present and evaluate two interaction techniques of Palmrest+: Palmrest Shortcut, designed for quick command execution, and Palmrest Joystick, designed for adjustment of continuous values.

Palmrest Shortcut is a method of executing commands using directional shear force input. When a user needs to perform a quick directionally related command, e.g., increasing font size and font alignment, the user can perform a subtle shear force input while maintaining the hands' home position for typing. Palmrest Joystick is a method for adjusting continuous values using continuous shear force control, particularly suitable for tasks requiring low precision control. For instance, when scrolling or resizing is needed while typing, users can adjust the values by seamlessly applying shear force on the palmrest area without switching their hands to the touchpad.

In this study, we initially developed a Palmrest+ prototype that detects shear force applied to the palmrest area while not modifying the laptop's existing form factor. We then conducted two preliminary studies, each aimed at thresholding the shear force profile to prevent false activation and accurately recognizing the intended direction of shear force among four distinct directions: up, down, left, and right. Afterward, we evaluated the performance of the Palmrest Shortcut for executing commands and assessed the performance of the Palmrest Joystick using a rate-control approach for target acquisition tasks.

The contributions of this work are as follows:

- We propose Palmrest+, a supplementary laptop input space that leverages shear force action on the palmrest area.
- Through empirical user studies, we verified the feasibility of the proposed concept, Palmrest+, with two input techniques,

Palmrest Shortcut for a rapid command execution and Palmrest Joystick for a subtle continuous value adjustment.

2 RELATED WORK

We first review the users' typical palm-resting behavior during laptop usage. Next, we review previous research or products that have aimed to address the same objective of Palmrest+, which is reducing the cost of physical movement during keyboard typing. Finally, as palmrest interaction utilizes force input, we review related work on utilizing shear force input.

2.1 Palm-Resting Behavior during Laptop Usage

The palmrest area is a crucial laptop element, providing a comfortable surface for users to rest their palms in a neutral posture while typing [6]. Supporting the palms during typing has been reported to decrease muscle tension in the shoulder [3], thus being preferred by users to rest their palms on it during typing.

Yim et al. [29] explored the palmrest as a space for haptic feedback. They found that users spend more than 90% of their time in a resting position while interacting with the keyboard. However, users frequently break their palm-resting state due to the following reasons. Firstly, while interacting with a touchpad, it was observed that the contact area of the dominant hand moved from the center to the lower part of the palmrest area. Secondly, the palm-resting duration of the left hand significantly decreased when pressing modifier keys, such as Ctrl, Alt, and Shift keys. This finding implies that users break their palm-resting state when they perform keyboard shortcuts with modifier keys. In sum, users do not comfortably rest their palms while interacting with the touchpad or triggering keyboard shortcuts. On the other hand, our proposed Palmrest+ enables users to comfortably rest their palms while supporting the conventional subtle inputs.

2.2 Reducing Cost of Switching from Keyboard to Touchpad

When typing on a laptop keyboard, users often need to switch to the touchpad for actions such as selecting menu items. The transition not only requires a physical movement from the keyboard to the touchpad but also disrupts the workflow and may decrease productivity [1, 7]. While keyboard shortcuts can be an alternative with lower physical switching costs, the large number of existing command-shortcut mappings made their memorization challenging [17]. Previous research has made various attempts to reduce these switching costs and enable intuitive command input. However, many approaches still necessitate large movement of the hand [5, 20, 31] or unnatural hand posture [32, 33], leading to another physical discomfort.

Previous studies have investigated the integration of touch [4, 24] or force sensing [2] into keyboard keys to facilitate seamless input, reducing the movement from the home typing position. However, these approaches require additional sensors for every key, which requires significant modifications to the existing laptop design and impacts the typing experience, as they share the same space used for text entry. In contrast, Palmrest+ only requires sensor augmentation on the two sides of the palmrest area.

Several studies have proposed more practical methods of command inputs while maintaining the home typing position on the keyboard. Keddisseh et al. [16] relocated a touchscreen next to the palmrest area, which displays a toolbar. Sindhwani et al. [25] proposed a technique that utilizes eye gaze for cursor positioning to reduce hand switching during text editing. However, these methods still require hotkey usage for activation.

2.3 Shear Force Input on Surface

Due to its potential for expanding the input space in a subtle way, shear force input has been extensively explored across various devices, including smartphone [12, 15, 26], computer display [14], tablet [13], and smartwatch [28]. Previous approaches suggested multi-dimensional input using normal and shear forces on touch screen [12, 14, 21, 26]. Lee et al. [18] evaluated the user controllability of shear force input on a mobile device screen. The results showed that people responded positively to shear force-based input. Harrison and Hudson [11] proposed the five classes of advanced interaction on touch screen utilizing the two-dimensional information inherent in the shear force. Huang et al. [15] suggested a more low-cost shear input interface on a touch screen using a transparent sheet and rubber band. Yu et al. [30] found that shear force-based input showed shorter completion times in 3-DOF and 6-DOF object manipulation tasks than the space mouse or touch interface.

While previous studies typically explored shear force interaction with fingertips, this study explores the shear force input exerted by the palms and assesses its performance compared with the conventional input method of the laptop.

3 PALMREST+

In this section, we introduce two interaction techniques of Palmrest+: Palmrest Shortcut and Palmrest Joystick. Palmrest Shortcut is designed for triggering a command rapidly, while Palmrest Joystick is designed for subtle continuous value input. We will describe each interaction method and use case examples.

We first define the term *Push* as the application of shear force on the palm exceeding a predetermined force threshold without a slippage. *Push* can be made in four directions: up, down, left, and right, which we call *Push Up*, *Push Down*, *Push Left*, and *Push Right*, respectively. *Release* is a term we define as the dropping of shear force below the threshold by finishing input action.

3.1 Palmrest Shortcut

3.1.1 Interaction Method. Palmrest Shortcut can be used by performing *Push* unimanually, bimanually, or in combination with a key press. These methods are referred to as *One-Hand Push*, *Two-Hand Push*, and *One-Hand Push+Letter*, respectively. Fig. 2 illustrates the three possible Palmrest Shortcut actions. For *One-Hand Push*, the user only needs to perform *Push* and *Release* with a single hand. For *Two-Hand Push*, the user must perform *Push* action with both hands before releasing the applied shear force. For both types of input actions, the associated command is triggered when one hand performs *Release* action. For *One-Hand Push+Letter*, the user sequentially presses a letter key on the keyboard after performing *Push* action. The associated command is triggered when the user presses the corresponding key while maintaining *Push* action. The

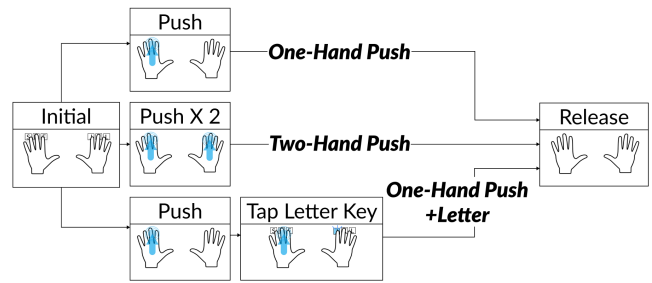


Figure 2: Interaction Flow of Palmrest Shortcut. User can perform *Push* action with a single hand (*One-Hand Push*) or with both hands simultaneously (*Two-Hand Push*). For *One-Hand Push+Letter*, the user needs to tap a letter key while maintaining *Push* action. The user finishes the input by *Release* action.

letter keys near the hand performing *Push* action are utilized in combination for the unimanual *One-Hand Push+Letter* action, or keys near the other hand that are not performing *Push* action are utilized in combination when performed for the bimanual *One-Hand Push+Letter* action.

One-Hand Push involves four directions with either the left or right hand, affording a total of eight possible commands. For the *Two-Hand Push*, although 16 (4×4) combinations of actions are possible, we recommend using only eight of these combinations, with both hands performing *Push* action either together on the horizontal axis or together on the vertical axis. This design recommendation is based on an in-lab pilot study, where users reported discomfort when performing a horizontal *Push* on one hand (e.g., *Push Left* with the left hand) and a vertical *Push* with the other hand (e.g., *Push Down* with the right hand). Lastly, for *One-Hand Push+Letter*, we defined the vocabulary with the combination of the direction of *Push* and the pressed key.

3.1.2 Use Case. There are numerous directionally related commands, including dichotomous commands (e.g., move previous/next, increase/decrease, open/close) or specific directional commands (e.g., align left/right, move up/down/left/right). Palmrest Shortcut may address the situation when a large amount of these directionally related commands need to be mapped to the shortcut. In such cases, keyboard shortcuts require numerous commands mapped to shortcuts using multiple modifiers or less intuitive letter keys, making it challenging to execute and memorize the shortcut [16]. Palmrest Shortcut is expected to expand a significant portion of shortcut vocabulary and allow the current keyboard shortcut design to become less crowded.

Palmrest Shortcut with *One-Hand Push* and *Two-Hand Push* are beneficial as they do not occupy letter keys and only require subtle movement of palm, allowing for additional shortcut vocabulary with their directional intuitiveness benefit. We suggest mapping *One-Hand Push* shortcuts to the commands related to directional movements, which frequently occur during typing. For instance, users can move the text cursor by word in a text editor, navigate cell by cell in a spreadsheet, or navigate through the objects in a whiteboard application (Fig. 3a). *Two-Hand Push* shortcuts with













(a) One-Hand Push	(b) Two-Hand Push with Same Direction	(c) Two-Hand Push with Opposite Direction
 <p>Move one word right (Text Editor)</p>	 <p>Select one word right (Text Editor)</p>	 <p>Zoom In</p>
 <p>Move one cell left (Spreadsheet)</p>	 <p>Extend selection by one cell left (Spreadsheet)</p>	 <p>Zoom Out</p>
 <p>Move one line up (Text Editor)</p>	 <p>Select one line up (Word)</p>	 <p>Redo</p>
 <p>Move to the object below (Whiteboard)</p>	 <p>Jump to the last object (Whiteboard)</p>	 <p>Undo</p>

Figure 3: Examples command mappings for One-Hand Push and Two-Hand Push Shortcuts. (a) *One-Hand Push* can be mapped to commands related to directional movement. (b) *Two-Hand Push* with both hands in the same direction can be mapped to selection or a larger movement. (c) *Two-Hand Push* with both hands in the opposite directions can be semantically mapped to zoom in/out and redo/undo.













(a) One-Hand Push + F keypress	(b) One-Hand Push + T keypress	(c) One-Hand Push + A keypress
 <p>Increase Font Size</p>	 <p>Open Tab</p>	 <p>Align Center</p>
 <p>Decrease Font Size</p>	 <p>Close Tab</p>	 <p>Align Justify</p>
 <p>Increase Font Size by 1 point</p>	 <p>Next Tab</p>	 <p>Align Right</p>
 <p>Decrease Font Size by 1 point</p>	 <p>Previous Tab</p>	 <p>Align Left</p>

Figure 4: Examples command mappings for One-Hand Push+Letter Shortcuts. (a) *One-Hand Push + F keypress* can be mapped to commands adjusting the font size, (b) *One-Hand Push + T keypress* can be mapped to commands related to tab operation. (c) *One-Hand Push + A keypress* can be mapped to commands for alignment.

both hands in the same direction can be mapped to a word-level text selection in a text editor and a cell selection in a spreadsheet application or moving to the first or last element in a whiteboard application (Fig. 3b). *Two-Hand Push* shortcuts with both hands in opposite directions can be semantically mapped the directional combinations to dichotomous commands such as zoom in/out and redo/undo (Fig. 3c).

Shortcuts with *One-Hand Push+Letter* intuitively expand shortcut vocabulary by allowing four directionally related commands mapped to a single letter, as the following examples show:

By signifying *Push Up* as increasing a lot (for something), *Push Down* as decreasing a lot, *Push Right* as increasing a little, and *Push*

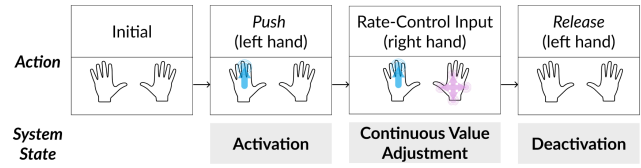


Figure 5: Interaction Flow of Palmrest Joystick. A user can activate the mode for Palmrest Joystick operation by maintaining *Push* action with the left palm and then perform the continuous value input in a rate-control manner by applying the shear force with the right palm. The user can deactivate the operation *Release* action with the left palm.

Left as decreasing a little, we can add the combination keypress for target context: *Push Up + F keypress* can be mapped to increase font size a lot (*Ctrl + Shift + >* in keyboard shortcut), and *Push Left + F keypress* can be mapped to decreasing font size by 1 point (*Ctrl + [* in keyboard shortcut) (Fig. 4a).

By signifying *Push Up* as opening (something), *Push Down* as closing, *Push Right* as going to the next (for something), and *Push Left* as going to the previous, we can add the combination keypress for target context: *Push Down + T keypress* can be mapped to closing a tab of an application window (*Ctrl + W* in keyboard shortcut), and *Push Left + W keypress* can be mapped to going to the previous window (*Alt + Shift + Tab* in keyboard shortcut) (Fig. 4b).

Directions of *Push* can be directly mapped to shortcuts containing directions, such as text alignment or inserting a row or column in a spreadsheet. *Align Center* is mapped to *Push Up + A keypress* (*Ctrl + E* in keyboard shortcut), and *Insert Columns to Left* is mapped to *Push Left + I keypress* (*Alt + i, then c, then c* in keyboard shortcut)

3.2 Palmrest Joystick

3.2.1 Interaction Method. Fig. 5 illustrates the interaction flow of Palmrest Joystick. First, there needs to be a distinct mode activation for Palmrest Joystick, which can be enabled either by maintaining a *Push* action in a certain direction with the non-dominant hand or additionally pressing a letter key after a *Push* action. Once activated, the user can manipulate continuous value with their dominant hand in a rate-controlled manner. The control speed is proportional to the applied shear force. Finally, the manipulation stops when the user stops maintaining *Push* action with the non-dominant hand.

We designed Palmrest Joystick as a rate-controlled input technique, as prior research has shown that shear force input is more suitable for rate control than position control [18]. In addition, the *Push* action of the non-dominant hand activates Palmrest Joystick to prevent false activation, thereby enabling the user to use the entire shear force space of the dominant hand. The decision to use the non-dominant hand for activating Palmrest Joystick while using the dominant hand for controlling a continuous value was based on Guiard’s Model of Bimanual Control [10]; the non-preferred hand is appropriate for performing coarse movements, while the preferred hand is suitable for fine movements.

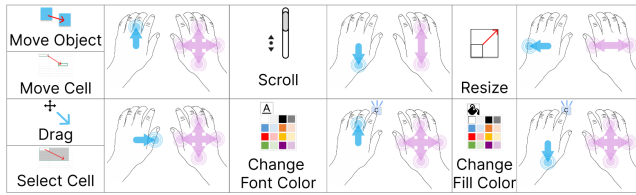


Figure 6: Use case examples for Palmrest Joystick. *Push Up*, *Push Down*, *Push Left*, and *Push Right* with the left hand can be mapped as the activation methods for moving, scrolling, resizing, and dragging operations. Additionally, *Push Up + C* keypress and *Push Down + C* keypress with the left hand can be mapped as the activation methods of font color adjustment and fill color adjustment, respectively.

3.2.2 Use Case. Although Palmrest Joystick may not offer a highly precise pointing up to the touchpad level performance, it can be advantageous in scenarios requiring frequent switching between brief continuous value input and typing, particularly when precise value control is not required. These tasks can be quickly operated by Palmrest Joystick without removing their hands from the home typing position.

Fig. 6 shows the example use cases of Palmrest Joystick. The user can perform *Push Up* to move the selected object in the whiteboard application, move the cell focus in a spreadsheet, or move the text cursor in the text editor. In addition, users may adjust the color of an object directly by activating the function by *Push Up* (font color) or *Push Down* (fill color) and combining it with the keypress of the C key with the left hand, while navigating through the color palette in 4×4 grid layout with the right hand.

4 PROTOTYPE IMPLEMENTATION

We built a proof-of-concept Palmrest+ prototype with a wireless keyboard, touchpad (Azoteq PXM0057-501-S), and two Palmrest+ modules, as shown in Fig. 7a. The Palmrest+ modules, depicted in Fig. 7b, is a force-sensing unit for the recognition of *Push* direction, consisting of a Lenovo TrackPoint¹, a Force Sensing Resistor (FSR), a vibrotactile actuator, and a microcontroller (Arduino Nano Every). Two palmrest modules are located on both sides of a touchpad. The width and depth of the prototype are matched to that of the Apple MacBook Pro 14 (5th Gen)².

The two-dimensional shear force is measured by the Trackpoint pointing stick, and a normal force is measured by the force sensor. The measured sensor values are sent to the PC through a USB serial communication at a sampling rate of 100 Hz. The vibrotactile actuator is modulated by a 250 Hz square wave with 5V for the haptic feedback on the palm for the "click" sensation when the user applies or releases the shear force.

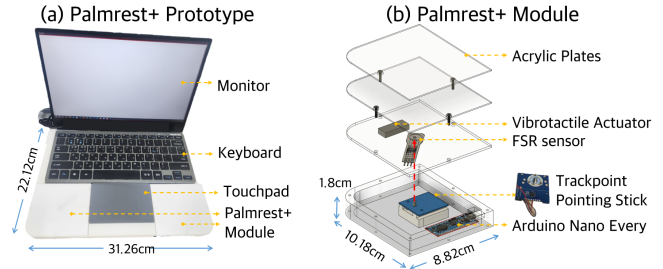


Figure 7: (a) Palmrest+ prototype with a portable monitor. (b) The inner structure of Palmrest+ module.

4.1 Activation Threshold of Shear Force for *Push* Action

To prevent unintended activation of Palmrest+ input, we determined the threshold for distinguishing intentional shear force input from normal resting behavior. To that end, we collected and analyzed the shear force range exerted by users on the palmrest area during their normal resting behavior.

Task and Instruction: Two tasks were designed to simulate typical laptop activities: text entry and spreadsheet application usage. In the text entry task, participants input 100 randomly selected sentences from the Mackenzie phrase set [19]. In the spreadsheet task, participants were instructed to replicate the task in a Microsoft Excel tutorial video³, involving creating tables, formatting styles, and generating charts. The study with two tasks took about 45 minutes. The participants were encouraged to perform the tasks as similar as possible to their natural laptop usage behavior.

Participant: We recruited 9 participants (5 male, 4 female), 19 to 25 years old (Mean = 21.2, SD = 2.3) from our university's online community. All participants reported that they use laptops almost every day. Participants were paid 15,000 KRW for their participation.

Result: Since the intentional occurrence of Palmrest+ input does not coincide with mouse or keyboard input, the shear force profile was collected when no mouse or keyboard inputs were active. Finally, thresholds for the *Push* action in each direction (up, down, left, and right) were decided as the 99.9th percentile highest values (excluding 0.1% highest force measured as an outlier screening). Fig. 8a shows the scatter plot of the shear force profile measured on left and right palmrest areas for all participants.

To convert the sensor value from the pointing stick into a Newton unit, we tested applying forces ranging from 1 to 5N on the palmrest module while measuring the corresponding sensor values. We utilized a linear translation stage to move a force gauge (IMADA DS2-20N) to apply the force while the palmrest module was fixed on the other side. Consequently, we found that the initial threshold for a certain orientation for a response was 0.88N, with a slope of 0.05N per sensor value. The sensor can detect a maximum of 7N for both horizontal and vertical axes. We finally determined the thresholds for activating shear force input on the palmrest in a Newton unit, as shown in Table 1.

¹Lenovo Trackpoint

²Apple MacBook Pro 14 5th generation

³Microsoft Excel Tutorial - Beginners Level 1 (<https://youtu.be/k1VUZEVDJ8>)

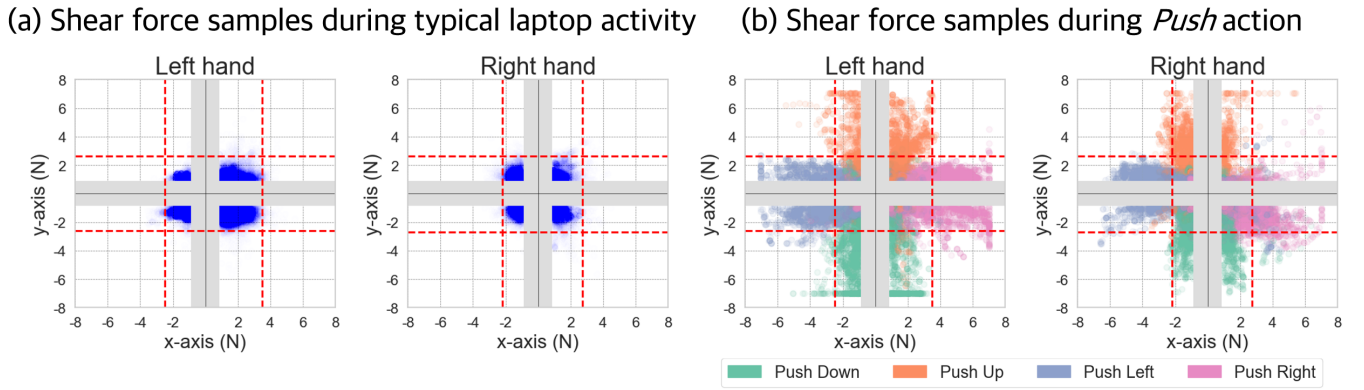


Figure 8: (a) Scatter plot of measured shear force samples when users perform typical laptop activities, including text entry and spreadsheet application usage. The detection threshold for each direction is indicated as the red dashed lines. (b) Scatter plots of measured shear force samples when users executed *Push* action for the four directions. The plots show that the shear force applied by users for a certain *Push* direction often exceeds the other direction’s threshold. (The gray areas indicate dead zones, which are below 0.88N, where the sensor cannot detect force due to the applied pressure being too weak.)

	Left	Right	Up	Down
Left hand	2.47	3.51	2.61	2.61
Right hand	2.22	2.71	2.61	2.71

Table 1: Threshold of shear force (Unit: N) used in *Push* action in four directions for each hand.

4.2 Recognition of the *Push* Direction

Accurately distinguishing the four directions of *Push* is crucial to ensure users’ intended input. Therefore, we observed the shear and normal force measured when intentionally performing *Push Up*, *Push Down*, *Push Left*, and *Push Right*. However, the classification between the four *Push* directions is found to be not accurately made solely by force thresholding due to overlapping force range profiles between different *Push* directions. Therefore, we decided to employ an additional classification process using a machine learning classifier.

Task and instruction: Each participant sat at a desk, placed their hands in their home typing position on the keyboard, and naturally rested their palms. After demonstrating *Push* actions for each direction, we instructed the participants to perform them. They attempted six trials for each of *Push* action in four directions with either left or right hands. Vibrotactile feedback was delivered when the user performed *Push* and *Release*, with the threshold determined from the earlier activation threshold study. Additionally, participants were required to type the keys on the middle row of the keyboard (f, d, s for left hand; j, k, l for right hand, in order) before and after each of the trials of *Push*, in order to position their hands in the home typing position.

Participant: We recruited 15 participants (8 male, 7 female), 20 to 30 years old (mean = 22.3, SD = 2.5) from our university’s online community. Participants were paid 20,000 KRW for their participation.

Result: We collected 720 data sets (15 participants \times 4 pivoting input \times 2 hands \times 6 trials) of the shear force and the normal force profile applied to the palmrest area for 100 Hz.

Fig. 8b shows the shear force profile while performing *Push Up*, *Push Down*, *Push Left*, and *Push Right*. The threshold of four directions of *Push* is marked by the red dashed line. From the observation, we found that the shear force applied by users for a certain direction of *Push* sometimes exceeds even the other direction’s threshold. When we solely decide the direction of the *Push* as the earliest detection matching, the detection accuracy is found to be 92.6%. This shows the limitation of relying solely on the thresholding approach, as it can lead to false activation of different directions of *Push*. Increasing the threshold of each direction may be one possible approach, but it would also increase the physical burden.

Therefore, we finally decided to employ classification through a Support Vector Machine (SVM) classifier. We used the adjacent 15 frames of shear and normal force as the input feature, specifically for instances when the intended force surpassed the threshold. We divided the dataset into training and testing sets to evaluate the model. For the evaluation of the classification accuracy, we conducted k-fold cross-validation with k=6, and the results reached 100% accuracy for all of the inputs.

In sum, *Push* is recognized when the threshold of a certain direction is exceeded, and the real-time classification result is the same as the detected direction. The maximum classification time taken was 4ms, which is not a perceivable delay by users.

5 EVALUATION OF PALMREST SHORTCUT

In this section, we report on a user study evaluating Palmrest Shortcut interaction. The primary focus is to test whether Palmrest Shortcut effectively reduces global hand movement during typing and minimizes the physical switching cost compared to keyboard shortcuts.

5.1 Study Design

The experiment followed a within-subject design with two independent variables: the type of shortcut (*ShortcutType*: *1-Modifier+Key*, *2-Modifier+Key*, *3-Modifier+Key*, *One-Hand Push*, *Two-Hand Push*, and *One-Hand Push+Letter*) and the block (*Block*: 1, 2, 3, 4, 5).

Three types of the Palmrest Shortcut (*One-Hand Push*, *Two-Hand Push*, and *One-Hand Push+Letter*) and three types of keyboard shortcuts (*1-Modifier+Key*, *2-Modifier+Key*, and *3-Modifier+Key*) were compared. Each keyboard shortcut consisted of the modifier keys and one letter key. The modifier keys were randomly selected from among those typically used in the Windows keyboard layout, including Ctrl, Alt, Shift, and Windows keys (e.g., Ctrl + q for *1-Modifier+Key* shortcut, Ctrl + Alt + p for *2-Modifier+Key* shortcut, and Ctrl + Alt + Shift + j for *3-Modifier+Key* shortcut). For each block, participants tried 8 shortcuts for each *ShortcutType* condition, 48 trials in total. For keyboard shortcut conditions, one of the four letters was randomly assigned from keys positioned on the left side (q, e, s, f, x, v), and four other distinctive letters from the right side (p, u, j, k, n, h) of the keyboard. For *One-Hand Push*, four directions of *Push* actions for each hand were tested. For *Two-Hand Push*, eight combinations in parallel directions of *Push* actions, as shown in Fig. 3b and 3c, were used. For *One-Hand Push+Letter*, four directions of *Push* actions for each hand were used with a combination of a letter key. This key was randomly chosen from either the left (q, e, s, f, x, v) or right (p, u, j, k, n, h) side of the keyboard, ensuring that four hand-key combinations were on the same side (e.g., left hand + q, right hand + p) and the other four combinations were on the opposite side (e.g., left hand + j, right hand + d).

5.2 Apparatus

The Palmrest+ prototype with a portable monitor was used for the testing environment as in Figure 7a. The keyboard layout was a standard layout for Windows. An eye tracker, Tobii Pro Nano, was attached to the bottom of the screen to record eye movements during the task. A webcam was placed above the palmrest prototype to record the hand movement.

5.3 Participant

We recruited 12 participants (8 male, 4 female) aged from 19 to 40 (mean = 24.4, SD = 6.3) from our university's online community. All recruited participants were right-handed and reported that they use the keyboard almost every day. All participants have experience using a Windows keyboard layout. Participants were paid 20,000 KRW for their participation.

5.4 Task

The task is designed to simulate a situation where users execute a shortcut in the middle of typing. A task trial consisted of three steps: the prior typing, the shortcut execution, and the posterior typing. During each task trial, the screen displayed the target shortcut to be executed for 1 second. For Palmrest Shortcut conditions, two hand images were shown to describe the required shortcut action, with an arrow upon the hands indicating the target *Push* direction. For *One-Hand Push+Letter*, a letter key was also shown next to the hand image, and participants were instructed to press the letter key with whichever hand they wanted. As soon as the target shortcut

illustration disappeared, participants had to type six letter keys (f, d, s, j, k, l, which are the letter keys on the central row of the keyboard) in sequential order. Following the prior typing of six letter keys, the illustration of the target shortcut was displayed again, and participants performed the shortcut. The keyboard shortcuts were detected when a letter key was pressed in combination with any of the modifier keys. Palmrest Shortcuts were detected when the shear force dropped below the detection threshold, i.e., *Release*. Lastly, they repeated typing the same six-letter keys again, and the task trial was completed.

After each trial completion, participants received feedback indicating whether the submitted shortcut was correct or not. Submitting a different shortcut was counted as a failure, and the same trial was repeated until the participant succeeded. Participants were instructed to complete each task trial as quickly and accurately as possible.

5.5 Procedure

After completing the demographic survey and consent form, participants were instructed on how to perform the task with each of six *ShortcutType* conditions using slides. Then, they went through the calibration process of eye trackers to collect gaze behaviors across conditions. There was a practice session of two trials for each *ShortcutType* condition. Then, participants performed 5 blocks of main testing, and each block consisted of 48 trials. Each block included 8 trials for each *ShortcutType*, in total 48 trials, presented in a completely randomly mixed order. In total, we collected data from 2,880 successful trials (12 participants \times 5 *Blocks* \times 6 *ShortcutType* \times 8 trials). After completing each block, participants rated a questionnaire, evaluating each *ShortcutType* condition's *Preference* and *Ease of Use* on a 7-point scale. In total, the experiment took approximately one hour to complete.

5.6 Performance Metrics & Analysis

We detected the hand landmarks to measure the maximal hand displacement by MediaPipe Hand Landmarker⁴, which exports 21 landmarks each in relative coordinates within the image. Since the dimensions of our prototype are known, we transformed the relative x and y-coordinates into world coordinates. We also collected eye gaze data to observe the number of keyboard glances during the task, using the Tobii Pro SDK for Python⁵

The dependent variables were as follows:

- *Execution Time*: the duration from the moment that the prior typing is completed to the moment the shortcut is triggered.
- *Return Time*: the duration from the moment that the shortcut is triggered to the moment that the first letter of the posterior typing is pressed.
- *Execution+Return Time (E+R Time)*: the total time spent on executing the shortcut and returning to typing, the summation of *Execution Time* and *Return Time*.
- *1st Success Rate*: the percentage of the shortcuts that participants correctly activated the shortcut in the first attempt.

⁴MediaPipe Hand Landmarker
(developers.google.com/mediapipe/solutions/vision/hand_landmarker)

⁵Tobii Pro SDK for Python
(developer.tobii.com/python/python-getting-started.html)

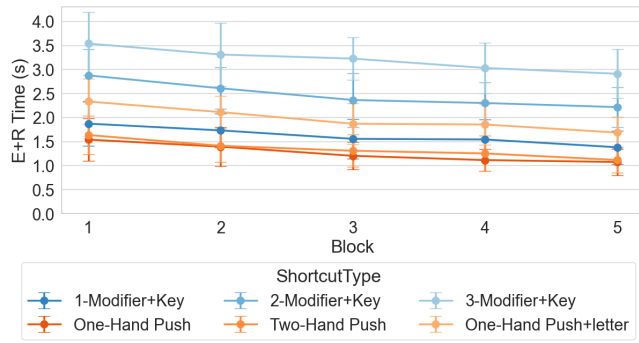


Figure 9: Linear plots of Execution+Return Time for ShortcutType per Block. Error bars show the standard deviation.

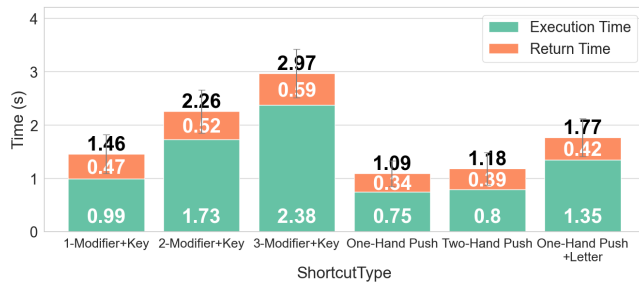


Figure 10: Stacked bar plot of mean Execution Time, Return Time for ShortcutType. Error bars show the standard deviation of Execution+Return Time.

- *Gaze Shifts*: the number of transitions from gazing at the monitor to the keyboard during the shortcut execution.
- *Max Hand Displacement*: the largest hand displacement occurred during the shortcut execution step. We set the reference hand position as the one at the beginning of the shortcut execution step, i.e., the completion of the prior typing step. Hand displacement was measured as the average distance from the center of each landmark point to the center of the corresponding reference landmark point, and the maximum value was picked.

One-way RM ANOVAs were used for metrics satisfying normal distribution (*Execution+Return Time*, *Return Time*, and *Max Hand Displacement*), and Friedman tests were used for metrics violating normal distribution (*1st Success Rate*, *Gaze Shifts*, and *Subjective Ratings*). For post-hoc comparisons, paired sample t-tests with a Bonferroni correction were performed for a parametric test, and a Wilcoxon Signed Rank test with a Bonferroni correction was performed for a non-parametric test. Corrected p-values are reported for the post-hoc tests.

5.7 Results

In this analysis, we mainly focus on the performance comparison between the keyboard shortcut conditions and Palmrest Shortcut conditions. Fig. 9 displays the *Execution+Return Time* over time, with the fastest completion times observed in the last two blocks.

Therefore, our subsequent analysis is based on data samples averaged from *Block 4* and *Block 5*.

5.7.1 Time. Fig. 10 shows the mean *Return Time* and *Execution Time* for each *ShortcutType* from trials where the target shortcut was correctly executed. The analysis showed *Execution Time* exhibited a similar tendency to the *Execution+Return Time*. Thus, we only reported the findings for the *Execution+Return Time* and *Return Time*.

A one-way RM ANOVA revealed significant effect of *ShortcutType* on *Execution+Return Time* ($F(5, 55) = 88.459, p < .001$). Post-hoc comparisons revealed that *One-Hand Push* (mean=1.09 s, SD=0.24) and *Two-Hand Push* (mean=1.18 s, SD=0.31) were significantly faster than *2-Modifier+Key* (mean=2.26 s, SD=0.40; $p < .001$ and $p < .01$, respectively) and *3-Modifier+Key* (mean=2.97 s, SD=0.46; both $p < .001$), but there were no significant differences with *1-Modifier+Key* (mean=1.46 s, SD=0.37). *One-Hand Push+Letter* (mean=1.77 s, SD=0.35) was significantly faster than the *2-Modifier+Key* ($p < .01$) and *3-Modifier+Key* ($p < 0.05$), but there was no significant difference with the *1-Modifier+Key*.

A one-way RM ANOVA revealed significant effect of *ShortcutType* for *Return Time* ($F(5, 55) = 16.677, p < .001$). Post-hoc comparisons revealed that *One-Hand Push* (mean=0.34 s, SD=0.12) shows faster *Return Time* than both *2-Modifier+Key* (mean=0.52 s, SD=0.12; $p < .05$) and *3-Modifier+Key* (mean=0.59 s, SD=0.12; $p < .01$), but there was no significant difference with *1-Modifier+Key* (mean=0.47, SD=0.12). *Two-Hand Push* (mean=0.39 s, SD=0.15) and *One-Hand Push+Letter* (mean=0.42 s, SD=0.10) show significantly faster *Return Time* than the *3-Modifier+Key* (both $p < .01$), but there was no significant difference with *1-Modifier+Key* and *2-Modifier+Key*.

5.7.2 1st Success Rate. Fig. 11a shows the average *1st Success Rate* for each *ShortcutType*. A Friedman Test revealed significant differences across *ShortcutType* for *1st Success Rate* ($\chi^2(5) = 20.104, p = .001$). Post-hoc comparisons revealed that none of the pairs showed significant differences. We could observe that the command execution with *One-Hand Push+Letter* condition showed the lowest *1st Success Rate* of 89%. We conducted a deeper analysis on it: the failure of the *One-Hand Push+Letter* execution mostly occurred by not pressing any key (55%) or pressing an incorrect letter key (14%) while the *Push* action was made correctly for target direction. The remaining cases (31%) were the cases where the *Push* action was made in the wrong direction.

5.7.3 Gaze Shifts. Fig. 11b shows the *Gaze Shifts* for each *ShortcutType*. A Friedman Test revealed significant differences across *ShortcutType* for *Gaze Shifts* ($\chi^2(5) = 56.779, p < .001$). Post-hoc comparisons revealed that *One-Hand Push* and *Two-Hand Push* showed significantly lower shifts to the keyboard than all the keyboard *ShortcutTypes* (all $p < .05$). *One-Hand Push+Letter* showed significantly lower shifts to the keyboard than *2-Modifier+Key* and *3-Modifier+Key* (both $p < .05$), but there was no significant difference with *1-Modifier+Key*.

5.7.4 Max Hand Displacement. To analyze reliable data samples from vision-based measurement, outlier samples were screened from the measured hand displacement dataset by excluding trials where *Max Hand Displacement* deviated larger than the 95% confidence level ($Z > 1.96$) from the mean for each left and right hand.

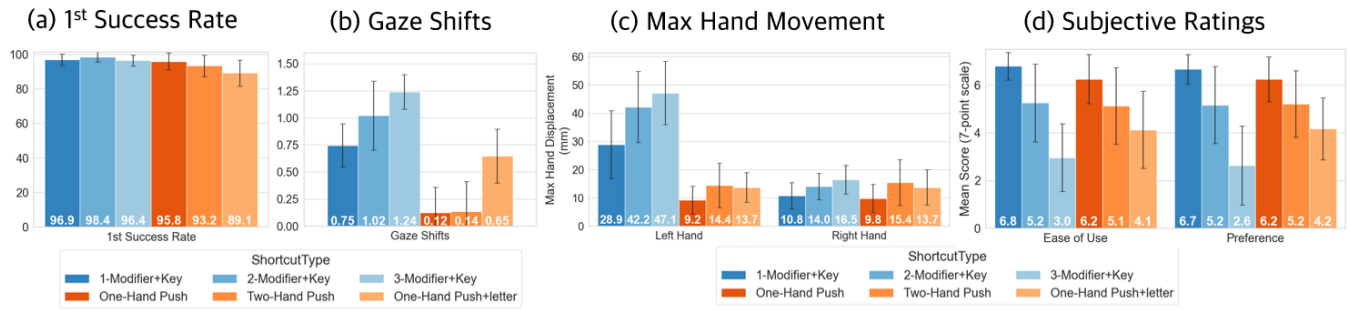


Figure 11: Bar plots of mean (a) *1st Success Rate*, (b) *Gaze Shifts*, (c) *Max Hand Displacement*, (d) *Subjective Ratings*. Error bars show the standard deviation.

The 111 and 142 outlier samples were removed for left and right-hand movements, respectively, accounting for 4.55% and 5.82% of the collected data. Fig. 11c shows the *Max Hand Displacement* of the left and right hand for each *ShortcutType*.

For the left hand, a one-way RM ANOVA test revealed significant differences across *ShortcutType* for *Max Hand Displacement* ($F(5, 55) = 53.382, p < .001$). Post-hoc comparisons revealed significant differences in all pairs except *One-Hand Push+Letter* and *Two-Hand Push*. In particular, *One-Hand Push*, *One-Hand Push+Letter*, and *Two-Hand Push* showed lower hand displacement than *1-Modifier+Key* ($p < .01$ for *One-Hand Push*, $p < .05$ for *Two-Hand Push* and *One-Hand Push+Letter*), *2-Modifier+Key* (all $p < .001$), and *3-Modifier+Key* (all $p < .001$).

For the right hand, a one-way RM ANOVA revealed significant differences across *ShortcutType* for *Max Hand Displacement* ($F(5, 55) = 6.067, p < .001$). Post-hoc comparison revealed that the *One-Hand Push* condition showed lower hand displacement than *3-Modifier+Key* ($p < .05$). Other pairs between keyboard *ShortcutType* and palmrest *ShortcutType* did not show a significant difference. Since the modifier keys are positioned on the left side, most hand displacement occurs on the left hand during the keyboard shortcut execution.

5.7.5 Subjective Rating. Fig. 11d shows the subject rating scores on a 7-point scale for each *ShortcutType*.

A Friedman Test revealed significant differences on *Ease of Use* rating across *ShortcutType* ($\chi^2(5) = 38.376, p < .001$). Post-hoc comparisons revealed that *One-Hand Push+Letter* was regarded as significantly more difficult than *1-Modifier+Key* ($p < .05$). *One-Hand Push* was regarded as significantly easier to use than *3-Modifier+Key* ($p < .05$), and *One-Hand Push+Letter* ($p < .05$). Other pairs of *ShortcutType* did not show a significant difference on *Ease of Use*.

A Friedman Test revealed significant differences on *Preference* rating across *ShortcutType* ($\chi^2(5) = 40.949, p < .001$). Post-hoc comparisons revealed that *One-Hand Push*, *Two-Hand Push*, and *1-Modifier+Key* were significantly preferable to the *3-Modifier+Key* ($p < .05$). *One-Hand Push* was significantly preferable to the *One-Hand Push+Letter* ($p < .05$). Other pairs of *ShortcutType* did not show a significant difference on *Preference*.

5.8 Qualitative Feedback & Summary

We interviewed participants after the study, asking, “What do you consider as the strengths and weaknesses of Palmrest Shortcut?”. Participants reported that the main benefit of Palmrest Shortcut is its “eyes-free” operation, saying that it reduces the need to look at the keyboard, which is common for keyboard shortcuts to find the target keys. P3 said “I do not have to check key positions for shortcut execution while executing *One-Hand Push* and *Two-Hand Push*”. P7 said: “Palmrest shortcuts were easier. When using keyboard shortcuts, I had to find the location of each finger”.

Regarding the *Two-Hand Push*, most participants favored the combination with opposite directions along the vertical axis. Some participants said performing *Push* action in the same direction on the horizontal axis was difficult (P1, P12). For the *One-Hand Push+Letter*, participants reported difficulties pressing certain keys while performing *Push* action with the same hand. P5 reported: “It was uncomfortable to press P after doing *Push Down* action with the right hand”, P3 mentioned: “It was easier to press the key with the other hand that is not performing *Push* action.”

Participants who are familiar with keyboard shortcuts tended to favor the *1-Modifier+Key* and *2-Modifier+Key* over Palmrest Shortcut. P6 said “*1-Modifier+Key* and *2-Modifier+Key* was easier since I use them every day and I am already used to them. *One-Hand Push+Letter* is least preferable since it still has to look at the keyboard.” Conversely, those less experienced with shortcuts or not proficient in blind typing tended to prefer utilizing Palmrest Shortcut. P12 noted “When using unfamiliar keyboard shortcuts, I had to first look at the position of modifier keys and the letter key on the keyboard. However, *One-Hand Push+Letter* was easier because I only had to remember the position of one letter key”.

Finally, we could confirm that the execution of Palmrest Shortcut induces much smaller hand displacements than keyboard shortcuts. Furthermore, Palmrest Shortcut was reported to be even faster than keyboard shortcuts with more than two modifier keys. The key factors contributing to this faster performance appear to be fewer gaze shifts toward the keyboard and reduced left-hand displacement during the shortcut execution, as supported by user interviews. In contrast, keyboard shortcuts demanded more gaze shifts toward the keyboard, particularly when executing unfamiliar combinations or those with multiple modifier keys.

6 EVALUATION OF PALMREST JOYSTICK

Palmrest Joystick can be beneficial for situations where frequent switching between typing and continuous value input occurs but when precise control is unnecessary. As the touchpad is highly optimized for precision pointing tasks, the goal of the study is not to outperform the touchpad but to explore Palmrest Joystick's effectiveness in these specific use cases where lower precision of control suffices. Therefore, we conducted a user study on a task integrating target acquisition tasks with low index of difficulty (i.e., small distance or large target size) in the middle of typing tasks.

6.1 Study Design

We aim to simulate a task situation where users execute continuous value adjustment in the middle of typing, such as zooming, panning, coarse focus moving, or scrolling. However, instead of testing all different types of tasks, we conducted a general 1D/2D target acquisition task typically used in Fitts' law experiments, with low Index of Difficulty. The experiment was designed as a within-subject with two independent variables: the control method (*Method*: Touchpad and Palmrest) and task type (*Task*: 1D and 2D). In a target acquisition task, we tested four distance conditions from the initial position to the target (distance = 120, 240, 360, 480px) and the four width conditions of the target (width = 60, 100px). The rectangle-shaped target was used in the 1D task, and the circle-shaped target was used in the 2D task, as shown in Fig. 12.

The 2D target acquisition task presented targets placed every 45° starting from 0° (straight upward). The 1D target acquisition task presented four targets, which were positioned on the up, down, left, and right sides from the origin.

6.2 Apparatus

The Palmrest+ prototype was used with a 20-inch screen display at 1920 × 1080 resolution, and a webcam was placed above the screen to record the hand movement.

6.3 Participants

We recruited 12 participants (6 males, 6 females) aged from 19 to 25 (mean = 20.8, SD = 1.8) from our university's online community. All participants were right-handed. Participants were paid 25,000 KRW for their participation.

6.4 Task

A task trial consisted of three steps: the prior typing, the execution of the target acquisition, and the posterior typing. Participants waited for 1 second to observe the target's position on the screen. Subsequently, they had to type 3 letter keys (j, k, l) in order with their right hand. As soon as the prior typing was completed, the pointing cursor returned to the screen center. The acquisition target was visualized in red. Participants were instructed to move the pointing cursor to the target using either a touchpad or Palmrest Joystick. When the cursor reached the target, the target's color turned green. In the Palmrest Joystick condition, the pointing cursor could be moved by applying shear force with the right palm, but only when the left hand was maintaining *Push* action. The selection of the target was made when *Release* action occurred with the left hand. In the touchpad condition, the selection was made by clicking,

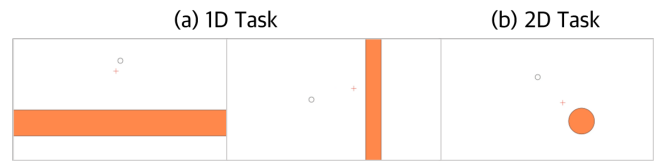


Figure 12: (a) 1D Task displays rectangular-shaped target that is aligned either in the horizontal or vertical axis and (b) 2D Task displays circular shaped target

i.e., pressing the surface. If the selection was made while the cursor was outside of the target, a beep was delivered. Note that, unlike the previous study, there was no case such as unsuccessful trials; the trial ended when the participant correctly pointed and selected the target. Once the target was selected, participants performed the posterior typing of 3 letter keys: j, k, and l. Participants were instructed to complete the task as quickly and accurately as possible. The CD gain of the cursor was predetermined for both the touchpad (12.15 px/mm) and Palmrest Joystick (0.28 px/N) through an in-lab pilot exploration.

6.5 Procedure

After completing the demographic survey and consent form, participants were instructed on how to perform the task with each condition: with touchpad and with Palmrest Joystick. Before the main testing session, participants had a practice session for about 10 minutes to familiarize themselves with each method and the task. Then, participants performed 3 consecutive blocks of tasks for each condition. Each block consisted of 64 trials, with 30 seconds resting between the blocks. The order of conditions was counter-balanced across the participants using a balanced Latin Square. After finishing the blocks, participants filled in the NASA-TLX questionnaires. In total, the experiment took approximately 1.5 hours to complete.

6.6 Performance Measures

The dependent variables were as follows:

- *Acquisition Time*: the duration from the moment that the pointing cursor began to move to the moment that the acquisition target was correctly selected. In the touchpad method, we measured from the moment when the cursor movement started, while in the palmrest method, we measured from the moment of the activation of *Push Up* action by the left hand.
- *Switching Time*: the summation of 1) the duration from the end of the prior typing until the beginning of the pointing cursor movement and 2) the duration from the completion of the target acquisition with either touchpad or Palmrest Joystick until the first keystroke of the posterior typing task.
- *Acquisition+Switching Time (A+S Time)*: the total time spent on executing the target acquisition with the touchpad or Palmrest Joystick and switching time from/to typing. It is the summation of *Acquisition Time* and *Switching Time*.
- *1st Success Rate*: the percentage of trials in which the user correctly selected the target from the first selection.
- *Max Hand Displacement*: the maximum hand displacement occurred during 1D/2D target acquisition task. We set the reference

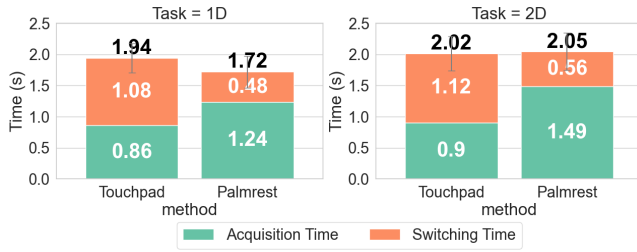


Figure 13: Stacked bar plot of mean *Acquisition Time* and *Switching Time* for *Method* \times *Task*. Error bars show standard deviation of *Acquisition+Switching Time*.

hand position as the one at the beginning of the target acquisition step, i.e., the completion of the prior typing step. Hand displacement was measured as the average distance from the center of each hand mark point to the center of the corresponding reference landmark point, and the maximum value was picked.

Two-way RM ANOVAs and paired t-tests with Bonferroni correction for post-hoc comparison were used for analyzing metrics satisfying normal distribution (*Acquisition Time*, *Switching Time*, *Acquisition+Switching Time*). For the metrics violating normality (*1st Success Rate*, *Max Hand Displacement*, NASA-TLX scores), we applied Aligned Rank Transform on the dataset [27] before conducting RM ANOVAs, and post-hoc pairwise comparisons with the ART-C was used [8].

6.7 Results

To ignore extreme data samples, we screened outlier data samples where the *Acquisition+Switching Time* was beyond 95% confidence interval range from the mean, for metrics related to time (*Acquisition+Switching Time*, *Acquisition Time*, *Switching Time*) and *1st Success Rate*. A total of 364 outliers were removed, representing 3.95% of the data collected.

6.7.1 Time. Fig. 13 shows the *Acquisition Time*, *Switching Time* for *Method* \times *Task* condition. For the *Acquisition Time* metric, a two-way RM ANOVA revealed that the palmrest method showed significantly slower than the touchpad method ($F(1, 11) = 97.409$, $p < .001$). However, for the *Switching Time* metric, a two-way RM ANOVA revealed that the palmrest method showed significantly faster than the touchpad method ($F(1, 11) = 236.566$, $p < .001$).

For *Acquisition+Switching Time*, a two-way RM ANOVA revealed that the main effect of the *Method* was statistically significant ($F(1, 11) = 7.274$, $p < .05$). The main effect of 1D and 2D task on *Acquisition+Switching Time* was statistically significant ($F(1, 11) = 20.479$, $p = .001$). There was a significant *Method* \times *Task* interaction effect on *Acquisition+Switching Time* ($F(1, 11) = 19.642$, $p = .001$). Post-hoc comparison revealed that Palmrest \times 1D (mean=1.72 s, SD=0.26) was significantly faster than Touchpad \times 1D (mean=1.94 s, SD=0.24; $p < .001$), Touchpad \times 2D (mean=2.02 s, SD=0.28; $p < .001$), and Palmrest \times 2D (mean=2.05 s, SD=0.30; $p < .01$).

In sum, Palmrest Joystick showed slower *Acquisition Time* but faster *Switching Time* than the touchpad method. In total, Palmrest Joystick showed faster *Acquisition+Switching Time* than the touchpad, particularly for the 1D target acquisition task.

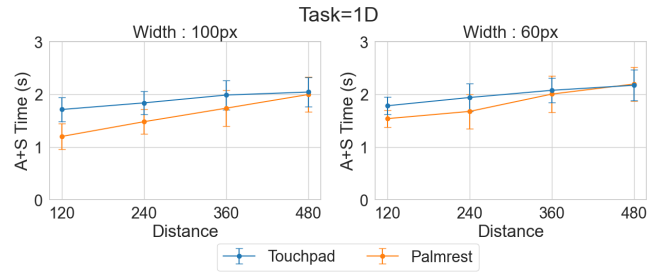


Figure 14: Line plot of *Acquisition+Switching Time* per width and distance for *Method* in 1D task. Error bars show standard deviation.

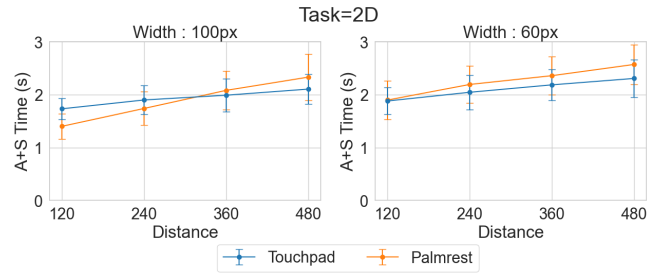


Figure 15: Line plot of *Acquisition+Switching Time* per width and distance for *Method* in 2D task. Error bars show standard deviation.

Additionally, we plotted the *Acquisition+Switching Time* by each distance and width condition. In the 1D target acquisition task, the palmrest method was faster for shorter distances due to the minimal switching cost. Still, it became similar to the touchpad method at a distance of 480px for both width conditions, as shown in Fig. 14. This observation shows that while the touchpad is faster than Palmrest Joystick for target acquisition, it requires longer switching time from and to the home typing position. Therefore, Palmrest Joystick became more efficient for short pointing movement during typing due to the minimal switching cost. In the 2D target acquisition task, Palmrest Joystick was faster than the touchpad only when the target width was 100px and the distance was less than 360px, i.e., large target size and small distance pointing. However, for all other width and distance conditions in the 2D target acquisition, Palmrest Joystick was slower than the touchpad, as shown in Fig. 15.

6.7.2 1st Success Rate. Fig. 16a shows the *1st Success Rate* for *Method* \times *Task*. Overall, the *1st Success Rate* of Palmrest Joystick was lower than that of the touchpad. We observed that when participants needed to make precise adjustments after moving the cursor close to the target, especially for a small acquisition target, several participants tended to stop the cursor movement by deactivating the left-hand *Push* action instead of releasing the shear force in the right hand, which led to lower *1st Success Rate* for Palmrest Joystick condition. However, they quickly activated Palmrest Joystick operation again by performing a quick left-hand *Push* action, and then made the correct target acquisition.

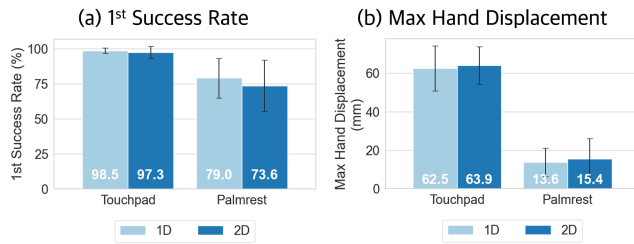


Figure 16: Bar plots of mean (a) 1st Success Rate, (b) Max Hand Displacement for Method \times Task. Error bars show standard deviation.

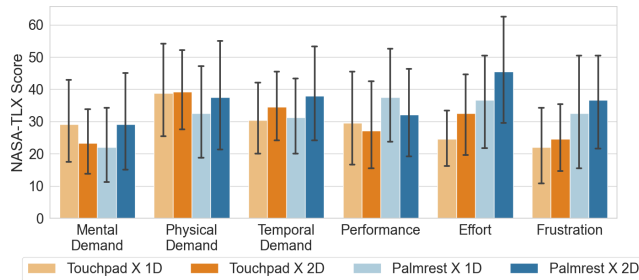


Figure 17: Bar plots of mean for Method \times Task NASA-TLX from Palmrest Joystick evaluation study. Error bars show standard deviation.

A two-way RM ANOVA revealed that the main effect of the *Method* on *1st Success Rate* was statistically significant ($F(1, 11) = 44.528, p < .001$), and the main effect of *Task* was statistically significant ($F(1, 11) = 13.1569, p < .01$). There was a significant *Method \times Task* interaction effect ($F(1, 11) = 9.9584, p < .01$).

6.7.3 Max Hand Displacement. As all the participants performed the touchpad operation with their right hand, we report the *Max Hand Displacement* for the right hand. To analyze reliable data samples from vision-based measurement, outlier samples were screened from the measured hand displacement dataset by excluding trials where *Max Hand Displacement* deviated larger than 95% confidence level ($Z > 1.96$) from the mean for the right hand. 226 outlier samples were removed for right-hand movements, accounting for 2.45% of the collected data. Fig. 16b shows the *Max Hand Displacement* for *Method \times Task*. Overall, the *Max Hand Displacement* of the right hand for the Palmrest Joystick was approximately four times lower than that of the touchpad. This somewhat anticipated observation clearly shows that Palmrest Joystick minimizes hand displacements, unlike the touchpad, which requires much larger hand movements to switch from the keyboard to the touchpad.

A two-way RM ANOVA revealed that the main effect of the *Method* on *Max Hand Displacement* of the right hand was statistically significant ($F(1, 11) = 51.3556, p < .001$). However, the main effect of *Task* ($F(1, 11) = 1.676, p = 0.222$) and *Method \times Task* interaction ($F(1, 11) = 1.107, p = 0.315$) was not statistically significant.

6.7.4 NASA-TLX. Fig. 17 shows NASA-TLX ratings for each *Method \times Task* condition. A two-way RM ANOVA revealed that there were

no significant effects of *Method* ($F(1, 11) = 1.397, p = 0.262$), *Task* ($F(1, 11) = 1.323, p = 0.274$), and *Method \times Task* interaction ($F(1, 11) = 1.271, p = 0.284$) on overall rating score.

6.8 Qualitative Feedback & Summary

We interviewed the participants after the study. We asked, “What are the benefits and limitations of Palmrest Joystick compared to the touchpad?”. Participants commented that the palmrest offers an advantage over the touchpad by reducing physical fatigue on the wrist (P1, P6, P9). P2 remarked, “*Fingers are more relaxed when using the palmrest method. In the touchpad method, I had to raise my finger and point at the touchpad.*” Similarly, P8 remarked, “*After typing and then using the touchpad, my wrist felt uncomfortable due to bending, but with the palmrest, my palm remains in contact, improving efficiency in movement.*” However, they also noted drawbacks of the palmrest method, such as difficulty in precise control (P1, P2, P6, P11, P12). P2 explained, “*Since the area of skin in contact is larger with the palmrest compared to fingertips, it was more challenging to control.*” Additionally, a couple participants mentioned that their palms got sweaty during the task, which they thought impacted their performance (P8, P10).

In conclusion, we analyzed the performance of the Palmrest Joystick and the touchpad across multiple criteria. Although the Palmrest Joystick showed slower acquisition time than the touchpad, its switching time from and to typing was quicker due to reduced right-hand displacement. Consequently, the sum of acquisition and switching time of Palmrest Joystick for one-dimensional targets that are small and in close distance was comparable to or even slightly faster than using the touchpad. However, precise control with the Palmrest Joystick was challenging, especially for small targets in two-dimensional conditions. Therefore, Palmrest Joystick is beneficial for manipulating one-dimensional targets or two-dimensional targets with large sizes and at close distances, especially when frequent switching between touchpad and keyboard occurs.

7 GENERAL DISCUSSION

7.1 Practicality of Palmrest+ Prototype

Our Palmrest+ prototype enabled the shear force sensing applied on the palmrest area by integrating a few sensors and actuators beneath the palmrest surface. This approach is a cost-effective solution without significantly altering the existing form factor of the laptop. To further ensure the practicality of the Palmrest+ technique, a discussion about the steps and costs involved in prototype development would be needed. For instance, the design of the stacked acrylic plates in a three-layer configuration requires careful consideration. These plates are not completely static but move slightly to apply a shear force to the Trackpoint sensor. They need to be rigid enough, and the movement should not be visibly noticeable while still allowing precise detection of shear force.

7.2 Controllability of Palm as an Input Technique

Input techniques using fingertips have been predominantly explored due to the high coordination of fingers. However, utilizing

the palm as an input modality remains underexplored, and most users have not experienced using the palm for input. We may further conduct a deeper analysis of palm controllability as an input technique. This could include longitudinal studies to assess the changes in performance over time and ergonomic considerations like hand fatigue. Additionally, we may investigate how different user groups, such as those with varying palm sizes, skin elasticity, and levels of sweatiness, adapt to palm-based input techniques and whether learning curves differ between them. These investigations could pave the way for developing more valuable input methods using palm.

7.3 Customization of Push Recognition

We observed significant variability in the force range applied to the palmrest during laptop activities among participants. To further enhance the accuracy of *Push* recognition, it would be beneficial to implement a personalization process involving adjusting the threshold and the classification parameters customized to individual users. Moreover, the palm resting behavior may depend on environmental factors, such as the position of the laptop. For instance, if the user is typing on a laptop that is placed on their lap, the force applied in the palmrest area can differ from when it is placed on a flat desk surface since the surface is slightly more tilted and lower positioned. We can further enhance the usability of Palmrest+ technique by fine-tuning the recognition of *Push*, considering the user's specific characteristics or the environment.

8 CONCLUSION

Palmrest areas of the laptop are consistently adopted in standard laptop design. Considering that users' palms are typically in contact with the laptop palmrest surface, it has the potential to be used as a supplementary input space. To explore the possibility of input on the palmrest area, we propose Palmrest+, which enables seamless and subtle input in the midst of typing the keyboard, with two interaction techniques utilizing shear force applied on the palmrest area: Palmrest Shortcut and Palmrest Joystick. Palmrest Shortcut facilitated rapid shortcut execution with minimized hand displacement from the home typing position, also in a more eyes-free way. Palmrest Joystick was found to offer minimized switching time from and to the keyboard compared to the conventional touchpad method. Overall, we confirmed that the idea of Palmrest+ has the potential to enhance user interaction with laptops by allowing subtle and seamless non-text input during typing.

ACKNOWLEDGMENTS

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2024-RS-2024-00436398) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation)

REFERENCES

- [1] Gilles Bailly, Eric Lecolinet, and Laurence Nigay. 2016. Visual Menu Techniques. *ACM Comput. Surv.* 49, 4, Article 60 (dec 2016), 41 pages. <https://doi.org/10.1145/3002171>
- [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* (Paris, France) (CHI '13). Association for Computing Machinery, New York, NY, USA, 563–572. <https://doi.org/10.1145/2470654.2470734>
- [3] Tom Bendix and Flemming Jessen. 1986. Wrist support during typing — a controlled, electromyographic study. *Applied Ergonomics* 17, 3 (1986), 162–168. [https://doi.org/10.1016/0003-6870\(86\)90001-3](https://doi.org/10.1016/0003-6870(86)90001-3)
- [4] Florian Block, Hans Gellersen, and Nicolas Villar. 2010. Touch-Display Keyboards: Transforming Keyboards into Interactive Surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) (CHI '10). Association for Computing Machinery, New York, NY, USA, 1145–1154. <https://doi.org/10.1145/1753326.1753498>
- [5] Daniel Buschek, Bianka Roppelt, and Florian Alt. 2018. Extending Keyboard Shortcuts with Arm and Wrist Rotation Gestures. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) (CHI '18). Association for Computing Machinery, New York, NY, USA, 1–12. <https://doi.org/10.1145/3173574.3173595>
- [6] Catherine Cook, Robin Burgess-Limerick, and Shona Papalia. 2004. The effect of wrist rests and forearm support during keyboard and mouse use. *International Journal of Industrial Ergonomics* 33, 5 (2004), 463–472. <https://doi.org/10.1016/j.ergon.2003.12.002>
- [7] R. F. Dillon, Jeff D. Edey, and Jo W. Tombaugh. 1990. Measuring the True Cost of Command Selection: Techniques and Results. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Seattle, Washington, USA) (CHI '90). Association for Computing Machinery, New York, NY, USA, 19–26. <https://doi.org/10.1145/97243.97247>
- [8] Lisa A. Elkin, Matthew Kay, James J. Higgins, and Jacob O. Wobbrock. 2021. An Aligned Rank Transform Procedure for Multifactor Contrast Tests. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) (UIST '21). Association for Computing Machinery, New York, NY, USA, 754–768. <https://doi.org/10.1145/3472749.3474784>
- [9] 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* (San Jose, California, USA) (CHI '07). Association for Computing Machinery, New York, NY, USA, 1591–1600. <https://doi.org/10.1145/1240624.1240865>
- [10] Yves Guiard. 1987. Asymmetric Division of Labor in Human Skilled Bimanual Action. *Journal of Motor Behavior* 19, 4 (1987), 486–517. <https://doi.org/10.1080/00222895.1987.10735426> arXiv:<https://doi.org/10.1080/00222895.1987.10735426>
- [11] Chris Harrison and Scott Hudson. 2012. Using shear as a supplemental two-dimensional input channel for rich touchscreen interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) (CHI '12). Association for Computing Machinery, New York, NY, USA, 3149–3152. <https://doi.org/10.1145/2207676.2208730>
- [12] Seongkook Heo and Geehyuk Lee. 2011. Force Gestures: Augmenting Touch Screen Gestures with Normal and Tangential Forces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) (UIST '11). Association for Computing Machinery, New York, NY, USA, 621–626. <https://doi.org/10.1145/2047196.2047278>
- [13] Seongkook Heo and Geehyuk Lee. 2013. Indirect shear force estimation for multi-point shear force operations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) (CHI '13). Association for Computing Machinery, New York, NY, USA, 281–284. <https://doi.org/10.1145/2470654.2470693>
- [14] Christopher F. Herot and Guy Weinzapfel. 1978. One-point touch input of vector information for computer displays. In *Proceedings of the 5th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '78)*. Association for Computing Machinery, New York, NY, USA, 210–216. <https://doi.org/10.1145/800248.807392>
- [15] Mengting Huang, Kazuyuki Fujita, Kazuki Takashima, Taichi Tsuchida, Hiroyuki Manabe, and Yoshifumi Kitamura. 2019. ShearSheet: Low-Cost Shear Force Input with Elastic Feedback for Augmenting Touch Interaction. In *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces* (Daejeon, Republic of Korea) (ISS '19). Association for Computing Machinery, New York, NY, USA, 77–87. <https://doi.org/10.1145/3343055.3359717>
- [16] Elio Keddissseh, Marcos Serrano, and Emmanuel Dubois. 2021. KeyTch: Combining the Keyboard with a Touchscreen for Rapid Command Selection on Toolbars. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) (CHI '21). Association for Computing Machinery, New York, NY, USA, Article 191, 13 pages. <https://doi.org/10.1145/3411764.3445288>
- [17] David M Lane, H Albert Napier, S Camille Peres, and Aniko Sandor. 2005. Hidden costs of graphical user interfaces: Failure to make the transition from menus and icon toolbars to keyboard shortcuts. *International Journal of Human-Computer Interaction* 18, 2 (2005), 133–144. https://doi.org/10.1207/s15327590ijhc1802_1
- [18] Boram Lee, Hyunjeong Lee, Soo-Chul Lim, Hyungkew Lee, Seungju Han, and Joonah Park. 2012. Evaluation of human tangential force input performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) (CHI '12). Association for Computing Machinery, New York, NY, USA, 3121–3130. <https://doi.org/10.1145/2207676.2208727>

- [19] I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems* (Ft. Lauderdale, Florida, USA) (*CHI EA '03*). Association for Computing Machinery, New York, NY, USA, 754–755. <https://doi.org/10.1145/765891.765971>
- [20] Hugh E McLoone, Ken Hinckley, and Edward Cutrell. 2003. Bimanual Interaction on the Microsoft Office Keyboard. In *INTERACT*, Vol. 3. 49–56.
- [21] Margaret R Minsky. 1984. Manipulating simulated objects with real-world gestures using a force and position sensitive screen. *ACM SIGGRAPH Computer Graphics* 18, 3 (1984), 195–203. <https://doi.org/10.1145/964965.808598>
- [22] H Moffet, M Hagberg, E Hansson-Risberg, and L Karlqvist. 2002. Influence of laptop computer design and working position on physical exposure variables. *Clinical Biomechanics* 17, 5 (2002), 368–375. [https://doi.org/10.1016/S0021-9290\(02\)00062-3](https://doi.org/10.1016/S0021-9290(02)00062-3)
- [23] Ramya K Prasad and Shanti Venkatesh. 2022. Understanding customer preferences on laptop variants and models for students and working professionals. *Academy of Marketing Studies Journal* 26, S4 (2022).
- [24] Yilei Shi, Haimo Zhang, Hasitha Rajapakse, Nuwan Tharaka Perera, Tomás Vega Gálvez, and Suranga Nanayakkara. 2018. GestAKey: Touch Interaction on Individual Keycaps. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) (*CHI '18*). Association for Computing Machinery, New York, NY, USA, 1–12. <https://doi.org/10.1145/3173574.3174170>
- [25] Shyamli Sindhvani, Christof Lutteroth, and Gerald Weber. 2019. ReType: Quick Text Editing with Keyboard and Gaze. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland UK) (*CHI '19*). Association for Computing Machinery, New York, NY, USA, 1–13. <https://doi.org/10.1145/3290605.3300433>
- [26] Taichi Tsuchida, Kazuyuki Fujita, Kaori Ikematsu, Sayan Sarcar, Kazuki Takashima, and Yoshifumi Kitamura. 2022. TetraForce: a magnetic-based interface enabling pressure force and shear force input applied to front and back of a smartphone. *Proceedings of the ACM on Human-Computer Interaction* 6, ISS (2022), 185–206. <https://doi.org/10.1145/3567717>
- [27] Jacob O Wobbrock, Leah Findlater, Darren Gergle, and James J Higgins. 2011. The aligned rank transform for nonparametric factorial analyses using only anova procedures. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 143–146. <https://doi.org/10.1145/1978942.1978963>
- [28] Robert Xiao, Gierad Laput, and Chris Harrison. 2014. Expanding the input expressivity of smartwatches with mechanical pan, twist, tilt and click. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 193–196. <https://doi.org/10.1145/2556288.2557017>
- [29] Jisu Yim, Sangyoon Lee, and Geehyuk Lee. 2023. HapticPalmrest: Haptic Feedback through the Palm for the Laptop Keyboard. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) (*CHI EA '23*). Association for Computing Machinery, New York, NY, USA, Article 153, 6 pages. <https://doi.org/10.1145/3544549.3585663>
- [30] Jinyang Yu, Jianjiang Feng, and Jie Zhou. 2023. PrintShear: Shear Input Based on Fingerprint Deformation. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 2 (2023), 1–22. <https://doi.org/10.1145/3596257>
- [31] 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* (Toronto, Ontario, Canada) (*CHI '14*). Association for Computing Machinery, New York, NY, USA, 1675–1684. <https://doi.org/10.1145/2556288.2557362>
- [32] Jingjie Zheng, Blaine Lewis, Jeff Avery, and Daniel Vogel. 2018. FingerArc and FingerChord: Supporting Novice to Expert Transitions with Guided Finger-Aware Shortcuts. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (Berlin, Germany) (*UIST '18*). Association for Computing Machinery, New York, NY, USA, 347–363. <https://doi.org/10.1145/3242587.3242589>
- [33] Jingjie Zheng and Daniel Vogel. 2016. Finger-Aware Shortcuts. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) (*CHI '16*). Association for Computing Machinery, New York, NY, USA, 4274–4285. <https://doi.org/10.1145/2858036.2858355>