Exploring Standard and Novel Interactions on a Multi-Touch-Enhanced Mouse

Mahmoud Sadeghi[a] and David Ahlström[b]
Interactive Systems, University of Klagenfurt, Austria
{*mahmoud.sadeghi, david.ahlstroem*}*@aau.at*

Keywords: Multi-Touch Surfaces, Mouse Interactions (Point-and-Click, Dragging), Window-Switching.

Abstract: The mouse is the primary pointing device in desktop computing. Its shape and form have advanced over decades, but its functionality is still limited to basic interactions. Inspired by early work, we envision multi-touch enhanced mouse input. With two user studies, we first show that our simple multi-touch mouse prototype – a smartphone attached to the base of a mouse – compares well to a standard mouse in point-and-click and dragging tasks. In a third study, we explore the potential of multi-touch enhanced mouse input and show that with Depth-Drag, a new multi-touch interaction, users are up to 47% faster when navigating and moving objects between stacked desktop windows than when they use a regular mouse.

1 INTRODUCTION

The computer mouse remains the primary input device for desktop computing, continually evolving through technical and ergonomic advancements to accommodate a broad range of tasks, from basic icon selection to complex image editing. Researchers have explored ways to enhance the mouse via hardware (Kim et al., 2020; Son et al., 2023) or software modifications (Ahlström et al., 2006; Grossman and Balakrishnan, 2005). Several projects introduced innovative designs that expand the mouse's input capabilities with additional sensors (Martin and Raisamo, 2004; Perelman et al., 2015; Villar et al., 2009; Shi et al., 2009; Yang et al., 2010; Saidi et al., 2017). However, most fail to convincingly demonstrate equivalence with the standard mouse in fundamental desktop interactions, such as point-and-click or dragging, thereby limiting their potential as viable alternatives.

Drawing inspiration from touchpads that allow for multi-finger gestures, we revisit the concept of having a multi-touch surface on a mouse. Benko et al. (2010) provide valuable insights into the design space of multi-touch mice, and the LensMouse (Yang et al., 2010) project inspired our prototype.

Our multi-touch mouse (MTM) prototype (Figure 1a) is straightforward: a regular smartphone attached to the base of a standard mouse. This enables multi-touch interactions via the touchscreen while allowing for exact cursor pointing via the mouse base. In two studies, we first demonstrate how MTM compares to a standard mouse in a point-and-click task and four different dragging tasks. After that, we introduce the Depth-Drag interaction on MTM (Figure 1b-d). Depth-Drag enables users to seamlessly move desktop items between stacked interface layers, such as between overlapping windows, content layers in graphic applications, or tabs in tabbed views. As such, Depth-Drag could serve to eliminate the often cumbersome rearrangement of windows, and it could also reduce reliance on the 'invisible' clipboard. In our third study, we evaluate the efficiency of Depth-Drag in a common file-moving scenario and compare its performance to the corresponding mouse-based interactions.

Our contributions are: 1) we present MTM, a multi-touch enhanced mouse prototype; 2) we devise and assess ways to perform the fundamental desktop interactions: point-and-click and dragging; 3) we introduce and evaluate Depth-Drag, a novel interaction tailored for touch-enhanced pointing devices, and 4) we sketch ideas and scenarios benefiting from multi-touch enhanced mouse-like devices, such as MTM.

2 BACKGROUND & PROTOTYPE

Given its importance, the computer mouse has been a focal point of substantial research. Previous studies have primarily aimed to 1) improve current mouse technology (ergonomics, hardware, software), 2) reduce reliance

[a] https://orcid.org/0000-0002-5475-7960
[b] https://orcid.org/0000-0002-9553-1685

Sadeghi, M. and Ahlström, D.
Exploring Standard and Novel Interactions on a Multi-Touch-Enhanced Mouse.
DOI: 10.5220/0013124200003912

In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2025) - Volume 1: GRAPP, HUCAPP and IVAPP*, pages 551-560
ISBN: 978-989-758-728-3; ISSN: 2184-4321

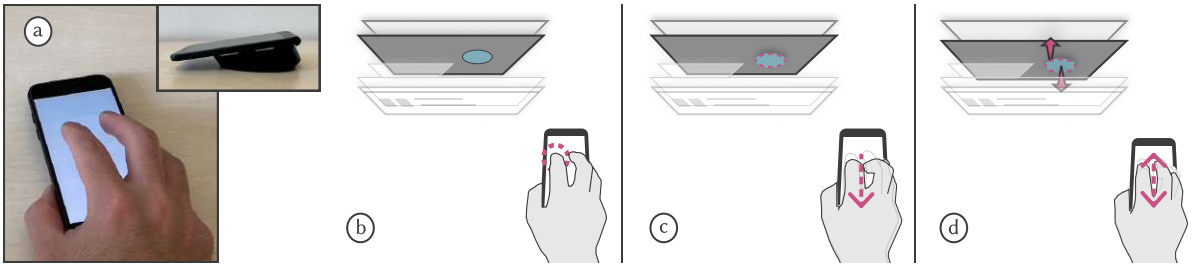

Figure 1: A) The MTM prototype combines cursor movements and touch gestures. With Depth-Drag interaction, the user quickly moves objects between stacked windows using b) a tap-click to select the source window, then c) a two-finger swipe to grab the desired object, and d) while holding it, two-finger swipe up or down to navigate to the destination window where the object is dropped using a finger liftoff.

on the mouse, or 3) enhance the mouse with extra hardware to provide add-on functionality. Researchers have investigated alterations to the mouse's physical characteristics, such as its shape (Peters and Ivanoff, 1999), size (Hourcade et al., 2007), weight (Conroy et al., 2022), or the position of its sensor (Kim et al., 2020), to assess their impact on pointing performance. Software-based solutions involve adjusting pointing targets on the screen (McGuffin and Balakrishnan, 2002) and optimizing cursor behavior (Ahmad et al., 2014; Casiez et al., 2008; Grossman and Balakrishnan, 2005). Researchers have also explored ways to remove the physical mouse by emulating its functionality via finger gestures on the computer's side (Mistry and Maes, 2011) or mid-air gestures above the keyboard (Taylor et al., 2014; Ortega and Nigay, 2009; Ultraleap, 2013). The third major theme in the mouse literature involves enhancing the mouse with additional hardware to broaden its functionality. Several projects have explored incorporating various sensors to add one or more extra degrees of freedom to the mouse. Examples include the VideoMouse (Hinckley et al., 1999), the TrackMouse (Martin and Raisamo, 2004), and the Gesture Mouse (Franz et al., 2016). Other projects have departed from mouse-like designs, presenting sophisticated prototypes with additional degrees of freedom, such as the GlobeFish (Froehlich et al., 2006), the Roly-Poly mouse (Perelman et al., 2015), and the TDome (Saidi et al., 2017).

Several projects have focused on adding touch sensors to the mouse. Hinckley and Sinclair (1999) presented two early prototypes and other mouse prototypes with touch-input have followed (e.g., Villar et al., 2009; Bartindale et al., 2010; Benko et al., 2010). Nowadays, a few commercial mice, such as Apple's Magic Mouse (2022) and Microsoft's Arc Mouse (2022) offer multi-touch sensing. However, these devices still rely on physical buttons for clicking and dragging. We focus on performing these primary interactions using touch gestures only. Researchers have also extended the mouse beyond a mere input device, using a touchscreen to provide both touch input and visual output (Yang et al., 2010; Saidi et al., 2017). We revisit the idea of a multi-touch mouse by employing a touchscreen-enhanced prototype, inspired by the work of Benko et al. (2010) and the LensMouse project (Yang et al., 2010).

Benko et al. presented one of the first studies of how to merge multi-touch input on a mouse with cursor movements into purposeful interactions. Their exploration revealed inherent challenges in designing effective multi-touch mouse interactions. Taking a visual route, Yang et al. introduce the LensMouse and demonstrate how offloading desktop content to its touchscreen for easy finger manipulation can speed up desktop work. While both Benko et al. and Yang et al. propose promising ideas for enhancing desktop interactions through multi-touch or touchscreen integration on the mouse, they do not address the execution of fundamental mouse interactions on such a device. Addressing this gap, our paper focuses on designing and evaluating point-and-click and drag interactions on a mouse-like device equipped with a touchscreen instead of mechanical click-buttons. We hope to save the similar mouse-related projects much of the laborious work in devising touch-input actions that translate to fundamental mouse interactions. Additionally, we seek to motivate future research by highlighting opportunities and use cases.

Our multi-touch mouse prototype (MTM) consists of an off-the-shelf smartphone atop a mouse base. The mouse base with its sensor provides precise cursor positioning, while the smartphone enables multi-touch input and visual output. We chose this simple approach over more complex methods, like mounting and wiring a lighter touch surface onto a mouse. We did not use devices such as Apple Magic Mouse or Microsoft's Arc Mouse, as they lack an SDK or supported method for developing interactions. For our first version, we used the base of a corded optical mouse (Logitech B100) and a Samsung Galaxy A3 smartphone (2016 model, Android OS version 7), mounted at a slight 'upward' angle facing the user (similar to the LensMouse). We

deactivated the hardware buttons at the bottom of the phone, and our software ignored touches within the bottom 5 cm of the screen to prevent unintended touch-events while gripping the device. We used this version in our first two studies on fundamental interactions (Study 1 point-and-click, Study 2 dragging). Informed by participants' feedback, we transitioned to a second version for our last study. Depicted in Figure 1a, the updated version has a wireless mouse base (LeadsaiL optical mouse) and a smaller smartphone (the 2017 Samsung Galaxy A3). More importantly, we mounted the phone at a 'downward' angle, slopping away from the user for a more comfortable grip and smoother finger gestures.

We conducted the point-and-click and dragging studies on a MacBook Pro (macOS version 11–13) connected to an external 27-inch Apple monitor (2560×1440 pixels). The laptop's monitor was turned off. We conducted the Depth-Drag study (Study 3) on the laptop's monitor (2560×1600 pixels, 13.3 inches diagonal) since smaller screens typically necessitate the layering of windows. We compared participants' task performance using the MTM prototype and a standard mouse, utilizing the same model as the base for MTM.

Our goal is not to exhaustively analyze every potential differences in participants' task performance between the mouse and MTM. Instead, Instead, our studies focus on fundamental point-and-click and dragging interactions as proof-of-concept tests, aimed at collecting data from tasks as realistic as possible and covering diverse desktop scenarios. Our analysis emphasizes the overall performance differences between the devices across various task variations.

3 STUDY 1: POINT-AND-CLICK

We start with point-and-click. Taking the familiar tap-action used on touchscreens, MTM has a Tap-Click to select: the user touches the MTM's screen with one finger ('mouse button' press) and swiftly lifts it off ('mouse button' release). We use duration and distance thresholds to distinguish a Tap-Click from the idle resting of one finger (or more) on the screen or from swipe gestures. Tap-Click is triggered if the duration between the finger-down and finger-up events is less than 200 ms and the touch movement is shorter than 1 mm. If more than one finger rests on the screen, Tap-Click must be executed with the leftmost finger for a left-click. For a right-click, at least two fingers must touch the screen, and the rightmost must issue the Tap-Click. Extensive experimenting helped us finalize these threshold values. Pilot tests informed us to use a short vibration (using the smartphone's vibration motor) to signal a successful click. We compared using MTM with Tap-Click to a standard mouse in point-and-click tasks.

3.1 Participants and Study Task

Twelve right-handed university students (4 female) aged 22 to 43 years (mean 32.6, *SD* 5.5) participated. Ten reported using a mouse, and two reported using a touchpad as their main pointing device.

Participants completed 120 point-and-click trials with both a mouse and Tap-Click in counterbalanced order. Each trial involved clicking inside a green start circle (diameter 16 mm), then moving the cursor to click inside a blue target circle. Target sizes (2, 4, 10, or 20 mm) and distances (75, 100, or 300 mm) were chosen to represent common macOS interface elements such as window buttons, dialog buttons, and application icons. The longest distance (300 mm) represented less frequent, long-range clicks (with study monitor ∼600 mm wide), while shorter distances (75 and 100 mm) more common, shorter-range clicks.

Timing started with a release event inside the start (i.e., mouse button release and finger-up with Tap-Click), and ended with a second release event anywhere outside the start. Start-target pairs were positioned, with the start to the left of the target in 50% of the trials (random order). We instructed participants to complete trials as fast and as accurately as possible.

Before the timed trials, participants adjusted preferred cursor speed in the macOS settings and performed 20 practice trials per technique. We enforced breaks between blocks of 12 trials (covering all size-distance combinations in random order). To start the next block, participants had to simultaneously press two keys at the opposite ends of the keyboard, marked with blue and red stickers. This mimicked the transition between keyboard and mouse use in typical desktop scenarios. A session lasted about one hour and ended with a short questionnaire, which included the participant's preference and rating of how easy it was to use each technique (from 1 to 10, 10 being very easy).

3.2 Results

The two techniques were on par regarding errors: overall, participants clicked outside the target in 78 trials (5%) when using the mouse and in 81 (6%) trials with Tap-Click.

To analyze the trial time, we removed all error trials. A successful trial lasted on average 1.01 s (*SD* 0.3) with the mouse and 1.27 s (*SD* 0.4) with Tap-Click ($t(11) = 13.30, p < 0.001$). This indicates that participants quickly adapted to MTM's alternative click

mechanisms with minimal practice. While Tap-Click had error rates similar to the mouse, it was slightly slower. Out of 12 participants, four preferred Tap-Click over the mouse. However, the mouse scored an average rating of 8.96 (*SD* 0.8), and Tap-Click scored 7.12 (*SD* 1.8). A Wilcoxon test showed that the difference was significant ($Z = -2.40, p < 0.05$).

4 STUDY 2: DRAGGING

Dragging with a mouse involves three steps: grabbing an object by pressing the mouse button, moving the mouse to the desired location, and releasing the button. The same steps apply to MTM, with two alternatives explored: Double-Tap-Hold (DTH) and Two-Finger-Swipe (TFS). DTH is similar to dragging on trackpads. To grab an object, the user makes a 'one-and-a-half' tap with one finger on the MTM surface; i.e., keeping the finger on the surface after the second touch. Like Tap-Click in Study 1, duration and distance thresholds identify the first tap in DTH. Exceeding the thresholds cancels DTH. After a correct grab, the user moves the cursor to the destination while keeping the finger on the touch surface. Lifting the finger releases the object. Multiple fingers can rest on the surface, but the leftmost finger must perform the actions.

With Two-Finger-Swipe (TFS), the user grabs an object by swiping downward with two fingers simultaneously and keeping them at the end position. Similar to Swipe-Click in Study 1, we employ swipe distance and direction thresholds to identify correct swipes in TFS. Any threshold violations cancel TFS. The user releases the object at the destination by lifting off both fingers from the surface. Lifting only one finger does not release the object.

Dragging serves various purposes. Some tasks, like moving files between folders, demand minimal precision. Others, such as adjusting slider knobs or highlighting text, require finer cursor control. High precision throughout cursor movement is crucial for tasks like tracing shapes in drawing applications. Our study incorporates three different tasks to cover these variations, while the fourth exemplifies combining touch-input actions into interactions not possible with a mouse and its buttons.

4.1 Participants and Study Tasks

Twelve right-handed university students (6 female) aged 22 to 40 years (mean 29.3, *SD* 4.8) participated. All used a mouse as their main pointing device. Four had prior experience using MTM from Study 1.

The box task in Figure 2a simulates situations

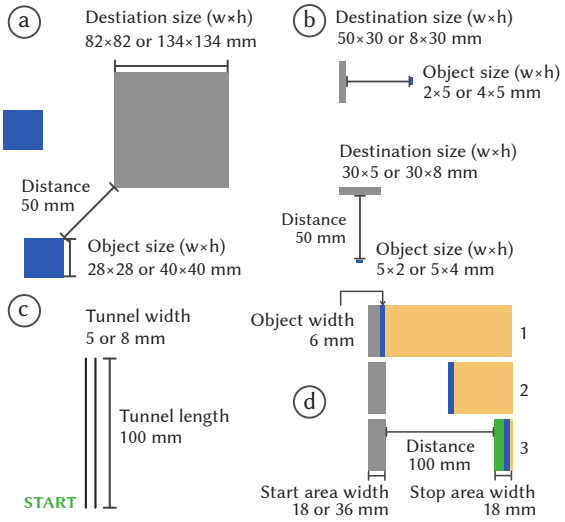

Figure 2: The four dragging tasks in the study: a) the box task, b) the bar task, c) the tunnel task, and d) the peek task.

requiring minimal precision during grabbing, dragging, and dropping, such as dragging icons between windows or moving windows. Participants move a blue square (representing the object) to a much larger gray destination area. The chosen object sizes (Figure 2a) represent a medium-sized file icon in macOS and the height of a floating video playback window (the largest draggable object we found in macOS). Movement is either straight or diagonal. The software randomly selects one of the intercardinal compass directions for diagonal trials and, for straight movement trials, one of the cardinal directions. The object-destination pair appears at random positions.

Timing starts when the cursor enters the object the last time before a grab event, and ends at object release while the cursor is inside the destination. Each block contains eight trials (2 object sizes $\times$ 2 destination sizes $\times$ 2 movement categories) in random order. Each participant completed 5 blocks for a total of 40 trials with each technique, mouse, DTH, and TFS (counterbalance order).

The bar task (Figure 2b) simulates situations that require precise object grabbing and dropping, such as aligning window borders or highlighting text. A small blue rectangle represents a narrow interface object (a window border or the space between words), and a gray rectangle represents an exact destination. The object sizes correspond to the spacing between words set in Arial 12pt and the size of a slider knob in macOS. To represent a high-precision task, the destination's constraining dimension is 1mm larger than the width or height of the largest object. The object-destination pair appears at a random position on the screen. Vertical trials had a 50mm distance, while horizontal trials had a coding error, resulting in a 73-78mm distance.

Timing begins when the cursor last enters the object before a grab event and ends when the object is released inside the destination. Incorrect trials (when the cursor is dropped outside the destination) are repeated randomly within the current block. Each block contains eight trials (2 object sizes $\times$ 2 destination sizes $\times$ 2 movement axes). Each participant completed 5 blocks, totaling 40 trials with each technique, mouse, DTH, and TFS (counterbalanced order).

The tunnel task (Figure 2c) represents scenarios where precision is required in cursor traversal, such as tracing objects in drawing applications. The tunnel is either horizontal or vertical and appears randomly on the screen. Timing starts when the cursor enters the tunnel and ends upon exit. Each participant completed five blocks of four trials (2 tunnel widths $\times$ 2 movement axes) with each technique, mouse, DTH, and TFS (counterbalanced order).

The peek task (Figure 2d) illustrates how multi-touch surface provides opportunities to devise new interactions for common desktop tasks. Users often need to temporarily resize windows to view content hidden behind or to see more of its content. To peek at content, the user grabs the window border and drags it until the desired content appears. At this point, the user can release the border to keep the new window size or drag the border back to its original location.

As shown in Figure 2d, the peek task consists of three steps: 1) the participant grabs the blue object (the window border), 2) drags it towards the opposite side of the yellow curtain (the window), which shrinks as the cursor moves, and 3) stops dragging when the green stop area becomes visible (revealing the content beneath). The participant then returns the blue object to the gray start area, either by dragging it back when using the mouse or by issuing an upward swipe with TFS or DTH. Timing starts when the cursor last enters the object before a grab event and ends when the object is released. The drag movement is either horizontal or vertical. The scene appeared at a random position on the screen. If the green stop area is never visible throughout the trial, the trial is repeated at a random position in the current block. Each participant completed five blocks of four trials (2 sizes of the stop area $\times$ 2 movement axes) with each technique.

We organized a session (lasting about one hour) in four parts. All participants began with the bar task, followed by the tunnel task, then the box task, and finished with the peek task, which required familiarization with the revert interaction. Before the timed trials, participants set their preferred cursor speed. Before beginning a new task type, we instructed the participant on performing the task with each technique. After that, the participant completed as many practice trials as needed to learn the current technique for the task. We counterbalanced technique orders among participants within each task type. After completing each task type, participants answered a short questionnaire and rated each technique's ease of use in that task (on a 1 to 10 scale, 10 being very easy).

4.2 Results

Table 1 presents the results for the dragging tasks. As in our previous analyses, we focus on an 'overall' comparison of the devices. For the time analysis, we removed the error trials and outliers in each unique task constellation. When a learning effect was observed with any of the techniques, we excluded the affected blocks for all techniques. We ensured the distribution of the remaining data was close to normal before applying an RM-ANOVA.

In the box task, the error rate was 0.5% (mouse), 4.7% (DTH), and 0.8% (TFS). We observed a learning effect in the first block for all techniques, hence excluded it from analysis. Error rates in the bar task were 4.4% (mouse), 7.5% (DTH), and 3.7% (TFS), with no learning effect observed. For the tunnel task, we excluded trials where participants crossed a tunnel border. This occurred in 19.2% (mouse), 20.7% (DTH) and 18.3% (TFS) of the trials. We saw the mean trial time in the first block with TFS was longer than in the other blocks, indicating a learning effect. The first block was excluded for all techniques.

The error rate in the peek task was 1.0% (mouse), 11.5% (DTH), and 7.2% (TFS). We noticed a longer mean trial time with DTH in the first block. We also saw decreasing times in the last two blocks with TFS, and a drop after the second block with the mouse. Thus, we only used data from the last two blocks. After RM-ANOVA (Table 1), Bonferroni post-hoc test showed only mouse and DTH significantly differ.

The results show that the mouse outperformed MTM techniques in the quick box task, which required minimal precision. While the mouse had a slight advantage in the precise bar task, the MTM techniques matched its performance in the tunnel task. The peek task, introducing a novel MTM interaction, had high error rates but was regarded as fun and practical by participants. With practice, users could become more comfortable with the reverting gesture and better integrate MTM interactions like peek into desktop workflows.

5 DEPTH-DRAG

Desktop and mouse users often have numerous windows open simultaneously, as many tasks require access to

Table 1: Analysis results of time and rating of the four dragging tasks.

Task	Metric	DTH	TFS	Mouse	Statistical Significance
Box	Mean Time	1.16 s (*SD* 0.2)	1.16 s (*SD* 0.1)	0.57 s (*SD* 0.1)	$F_{2,22} = 248.69, p < 0.001, \eta^2 = 0.96$
	Mean Rate	8.25 (*SD* 1.4)	8.50 (*SD* 0.8)	9.75 (*SD* 0.6)	$\chi^2(2) = 11.53, p < 0.05$
Bar	Mean Time	2.1 s (*SD* 0.4)	2.3 s (*SD* 0.3)	1.4 s (*SD* 0.2)	$F_{2,22} = 4.35, p < 0.05, \eta^2 = 0.75$
	Mean Rate	7.50 (*SD* 1.7)	7.33 (*SD* 1.6)	9.45 (*SD* 0.8)	$\chi^2(2) = 12.04, p < 0.05$
Tunnel	Mean Time	1.50 s (*SD* 0.8)	1.50 s (*SD* 0.7)	1.52 s (*SD* 0.9)	No significant difference found
	Mean Rate	8.08 (*SD* 1.4)	7.75 (*SD* 1.7)	8.83 (*SD* 1.3)	No significant difference found
Peek	Mean Time	2.22 s (*SD* 0.6)	2.17 s (*SD* 0.5)	1.96 s (*SD* 0.4)	$F_{2,22} = 5.77, p < 0.05, \eta^2 = 0.34$
	Mean Rate	8.5 (*SD* 1.0)	7.8 (*SD* 1.2)	9.4 (*SD* 0.9)	No significant difference found

various applications and their content. Studies have shown that having eight or more windows open is common and that window-switching is frequent and time-consuming (Smith et al., 2003; Hutchings et al., 2004; Warr et al., 2016). Ideally, with all windows visible, window-switching becomes a trivial point-and-click task. However, limited screen space, particularly on laptops, often leads to overlapping or maximized windows stacked in layers. This makes direct switching with a single click impossible.

Most desktop systems provide dedicated window-switching mechanisms, such as keyboard shortcuts (Alt/Cmd+Tab), taskbars with icons for open applications (or windows), and hotkeys or 'hot screen corners' that display clickable proxies for open windows. The importance of effective window-switching is evident in extensive previous research on novel approaches to window-switching (e.g., Smith et al., 2003; Warr et al., 2016; Xu and Casiez, 2010; Tak et al., 2009, 2011; Sato et al., 2020) and window management (e.g., Tashman, 2006; Ahlström et al., 2009; Zeidler et al., 2013; Tashman and Edwards, 2012). Here, we see opportunities for a multi-touch enhanced mouse and designed an MTM interaction for situations involving copy- or cut-and-paste content between windows. With many windows open, such situations often demand extensive window-switching (to find the desired windows) or cumbersome re-arrangement of windows (to make the content visible). Typical scenarios include copying text from a web page to a text document, pasting cell content between spreadsheets, and moving files between file browser windows.

Depth-Drag is simple. It partly relies on the taskbar and works as follows. Assuming a file-moving scenario where the user wants to copy files from different file browser windows (i.e., folders) into one common destination folder (open in its own file browser window). All windows are maximized and stacked, with the destination window at the top of the stack. The user first moves the cursor to click (with a tap on the touch surface) the desired window icon in the taskbar to activate the corresponding file browser window, which hosts the desired file. With the window active, the user moves the cursor to the desired file icon and 'grabs' it by moving two fingers simultaneously in a short downward swipe gesture over the touch surface. At the end of the swipe, when the fingers depart from the touch surface, the second window in the stack (i.e., the file browser window displaying the destination folder) is automatically brought to the front, and the grabbed file is dropped inside.

A second version of Depth-Drag uses MTM's visual output capability (through its touchscreen). We call it Depth-Drag-Visual. With the visual version, the user taps with two fingers on the MTM to display a grid with the window icons from the taskbar on the MTM's touchscreen, as visualized far right in Figure 3. A tap on the desired icon forwards the corresponding window to the front, and the user can continue the task the same way as when using the non-visual Depth-Drag. This way, the user can save long cursor movements to and from the taskbar.

The Depth-Drag interaction also allows for navigation through layers of stacked windows – hence, its name. If the user briefly pauses after the downward two-finger swipe and slides them upwards without lifting off the fingers, hidden windows in the stack appear up front, one replacing the other as the user slides further. Sliding down again brings back the previously shown windows, one after the other, while the user continues downwards.

6 STUDY 3: DEPTH-DRAG

We invited 12 persons (6 female) to participate in a study that compared Depth-Drag, Depth-Drag-Visual, and a standard mouse in a file-moving task, similar to the above-mentioned description. They aged 24 to 56 years (mean 36.3, *SD* 8.4). All used a mouse as their main pointing device. Four had used MTM in one of the earlier studies.

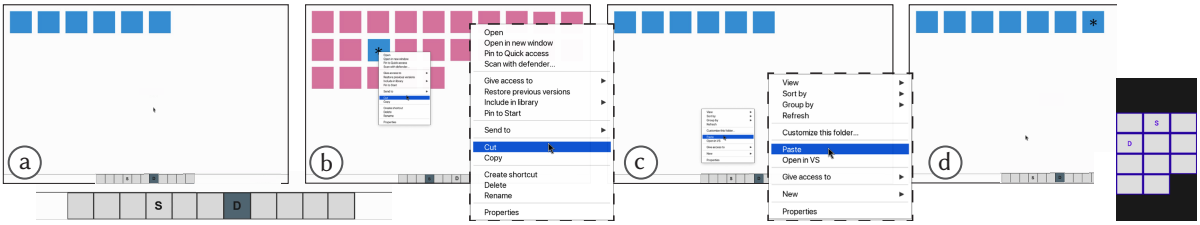

Figure 3: To cut-and-paste with the mouse in Study 3, the user: a) clicks on the 'S' icon in the taskbar to activate the source window, then b) right-clicks on the target item and selects 'Cut' from the context menu, then clicks on 'D' icon to activate destination window, and c) right-clicks and selects 'Paste', which d) moves the target item to the destination window.

6.1 Study Task and Procedure

Before each trial, the study software displays a "Start Trial" button in a maximized window. Clicking the button loads the first task screen and starts timing. Figure 3 depicts the remaining task screens. The taskbar at the bottom displays 11 icons, with windows' icons 'D' (destination) and 'S' (source) in random positions. The first screen (Figure 3a) shows the maximized destination window containing several identically colored square items (number and color vary per trial). The participant needs to navigate to the source window to cut a target item (marked with a star) and paste it back into the destination window. With the mouse or Depth-Drag, the participant clicks (one-finger tap in Depth-Drag) on the 'S' icon in the taskbar. When using Depth-Drag-Visual, the taskbar is hidden. The participant taps with two fingers on the MTM touchscreen to display a grid representation of the taskbar (Figure 3, far right). A one-finger tap on the 'S' icon activates the source window (Figure 3b). It contains 5 to 45 items arranged in a grid (the number varied randomly between trials; their size always 22×22 mm).

With the Depth-Drag techniques, participants activate the source window, move the cursor to the target item, perform a two-finger downward swipe to 'select' it, and finish by lifting off both fingers. The liftoff activates the destination window and pastes the 'grabbed' item, ending the trial (and timing). With the mouse, the task involves using the context menu to cut and paste the target item. We acknowledge that expert users often cut and paste using keyboard shortcuts. However, we aim to strictly compare the two pointing devices.

Any diversion from this sequence of actions (e.g., selecting the wrong icon in the taskbar or grabbing the wrong item in the source window) marks the trial as an error and ends it. Error trials were reloaded at random positions within the current block.

Each participant completed two series of three trial blocks, each ending after 20 error-free trials. A series included one block per technique, with the order counterbalanced across participants. Participants received instructions and practice trials with all techniques before starting with the first series of timed blocks: Tech-1 (practice+trials), Tech-2 (practice+trials), and Tech-3 (practice+trials). Participants answered NASA TLX (Hart and Staveland, 1988) questions for each technique after its block of trials in the second series: Tech-1 (trials+NASA-TLX), Tech-2 (trials+NASA-TLX), and Tech-3 (trials+NASA-TLX).

6.2 Results: Error Rate and Trial Time

Participants made errors in 54, 87, and 86 trials with the mouse, Depth-Drag, and Depth-Drag-Visual, resulting in error rates of 10.1%, 15.3%, and 15.2%, respectively. We removed error trials from subsequent analyses.

Participants could quickly pick up both Depth-Drag techniques. However, they were notably slow in the first timed trial with each technique and in Trial 21 (starting the second series) due to adjusting to another technique. Hence, we excluded Trial 1 and Trial 21 from all participants in our time analyses and confirmed the approximate normality of the remaining data.

The overall mean times were 6.67 s (*SD* 0.5) with the mouse, 3.52 s (*SD* 0.3) with Depth-Drag, and 4.59 s (*SD* 0.6) with Depth-Drag-Visual. An RM-ANOVA revealed a significant effect for technique ($F_{2,22} = 219.50, p < 0.001, \eta^2 = 0.95$), with Bonferroni-adjusted post-hoc tests confirming significant differences between all pairs (all p's < 0.001). Participants were 23% faster with Depth-Drag than with Depth-Drag-Visual, and they were 47% faster with Depth-Drag compared to the mouse. They were 31% faster with Depth-Drag-Visual than with the mouse. While we anticipated better performance with the Depth-Drag techniques, the magnitude of the improvement exceeded expectations.

6.3 Workload Ratings

Comparing our participants' subjective workload ratings across the six NASA-TLX dimensions–mental demand, physical demand, temporal demand, effort, frustration, and performance– revealed no significant differences for most dimensions among the three techniques. However,

a Friedman test showed a significant difference in the mental demand ratings ($\chi^2 = 7.70, p < 0.05$). Post-hoc Wilcoxon signed-rank tests with Bonferroni adjustment identified the difference between Depth-Drag-Visual and the mouse ($Z = -2.54, p < 0.05$), with Depth-Drag-Visual receiving higher mental workload ratings. This result resonates with participants' feedback and comments during the study. Almost all participants noted that frequently switching visual attention between the desktop monitor and MTM's touchscreen was mentally taxing: glancing at the 'mouse hand' felt unnatural and challenging to remember. A few even stated frustration about the need to shift focus constantly. This opinion, however, did not manifest itself in the NASA-TLX frustration ratings.

7 DISCUSSION

We first discuss and highlight our three studies' main results and conclusions, along with thoughts on improving the tested MTM interactions. After that, we present our ideas regarding possible future routes for our multi-touch-enhanced mouse.

7.1 Findings

Given the mouse's unmatched ubiquity, the results of our first two studies on fundamental point-and-click and dragging interaction did not surprise us.

Point-and-click is the most basic and rapid mouse interaction and an office worker points and clicks countless times during a workday. Accordingly, there is not much room for improvement. However, the Tap-Click we used in Study 1 is also rapid, and many people are skilled in tapping on their smartphones. We suspect that a more ergonomic multi-touch device than our MTM prototype can be as effective in point-and-click with a mouse.

In the box and bar dragging tasks, the mouse had several specific advantages that contributed to its significant performance lead over MTM. First, the point-and-click study revealed that the Tap-Click gesture on the MTM did not match the mouse click. Expectantly, an MTM gesture (Double-Tap-Hold), more complicated than Tap-Click, performs much worse than the simpler mouse technique (press to grab). Second, grabbing with MTM required a short mental pause from participants to perform the correct grab technique, while pressing the mouse button to grab is the least demanding action on the mouse. Lastly, the mouse allowed the users to grab the object while passing over it, a capability not replicated with MTM. The tunnel task results demonstrate that in careful maneuvering of the two devices, the difference in the weight or the shape did not influence the time.

The results of the peek task were the most unexpected, as we suspected the revert gesture provided a clear advantage to MTM. Upon closer examination, we found that the revert gesture in Double-Tap-Hold introduced additional cognitive load compared to Two-Finger-Swipe. While swiping up to revert in Two-Finger-Swipe aligns with the natural movement of the fingers, performing the same gesture in Double-Tap-Hold requires a different cognitive process. Additionally, the physical positioning of the fingers during the revert gesture posed challenges for participants, who expressed this issue with statements such as "The revert gesture was not intuitive" and "It was confusing whether to swipe up or lift the active finger." The results did not show a difference between the mouse and Two-Finger-Swipe, and we believe performance with this technique can reach its potential after more practice and familiarization with the revert gesture. Our observation strengthens this in that some participants paused briefly in the stop area to recall the revert gesture.

Participants were 31% faster with Depth-Drag-Visual than with the mouse. We expected to see better performance with the two Depth-Drag techniques than with the mouse. The MTM provides a direct drag using one gesture, while the mouse requires two steps. However, we did not expect such large differences.

The notable time advantage of MTM-Non-Visual over MTM-Visual is intriguing. Since both MTM techniques employ identical dragging gestures, the time difference stems from the windows-switching methods. Although the MTM-Visual saved the travel to the taskbar, the special gesture to activate the task grid, looking at the MTM screen, and selecting an icon significantly impeded participants. Participants also expressed discomfort and annoyance with the MTM-Visual due to the constant need to switch visual focus between the monitor and the MTM screen.

With our studies, we have shown that it is possible to combine sequences of touch gestures performed on top of a mouse into interactions mapped to the fundamental desktop interaction, point-and-click and dragging. Our Depth-Drag interaction also demonstrates that sequences of multi-touch gestures can be tailored into new interactions that are impossible to perform with a standard mouse and its two buttons.

7.2 Future Work

Our ideas for future work include exploring 'bounded' touch interactions, scroll and pan-and-zoom functionalities, and mode switching.

Bounded Touch Interactions. The current MTM interactions we tested can all be performed at any position on the screen (excluding the bottom part, which we deactivated to avoid unintended contact with the palm of the hand). Nevertheless, having invisible tapping areas on the surface opens up yet another design space for a multi-buttoned mouse with customizable functionalities for the buttons. This idea expands on the LensMouse's (Yang et al., 2010) feature to host visual contextual buttons but without the drawback of forcing the user to look at the visual display on the pointing device (which contributed to the mediocre performance of Depth-Drag-Visual in our third study). We plan to investigate the use of invisible buttons and how accurately users can tap different areas on the screen without looking. We predict seeing good results since people are good at relying on their muscle memory. Leveraging the touch surface's physical boundaries is another promising way to further expand the input vocabulary with interactions starting from the edge of the surface, commonly seen on smartphones and smartwatches (Roth and Turner, 2009; Serrano et al., 2013; Ahn et al., 2018).

Scroll, Pan, and Zoom. We plan to study scrolling, panning, and zooming with our multi-touch mouse. These interactions are commonly used for navigating content, such as text documents, maps, or images. Scroll, pan, and zoom can be designed as multi-touch gestures performed anywhere on the MTM touch surface. Moreover, we see great potential in binding these interactions to specific regions or edges of the surface. For instance, while vertically swiping along the edge of the MTM could invoke zoom, swiping elsewhere could scroll. We also envision using surface regions to configure interactions: swiping on the left side enables quick long-distance scrolling (using a high-gain transfer function), whereas swipes on the right side are used for high-precision positioning (using a function with a low gain).

Mode Switching. Multi-touch capabilities provide another area for exploration. We envision utilizing gestures performed with a secondary finger, such as the middle finger to switch between modes while performing interactions with the index finger. For example, in a spreadsheet application, the user can swipe 'in' from the right bezel with the middle finger to copy the formula in a cell. While holding the finger on the screen, the user can move the cursor to different cells and repeat a paste gesture with the index finger.

8 CONCLUSION

We have presented a multi-touch-enhanced mouse prototype. Through two user studies, we have demonstrated that our prototype, a smartphone attached to the base of a mouse, performs comparably to a standard mouse in point-and-click and dragging tasks where the user issues touch gestures instead of clicking with hardware buttons. Our third study highlights the potential of joining multi-touch with precise cursor positioning. Through a combination of finger movements on the touch surface, the Depth-Drag interaction allows users to quickly move objects between layers of desktop windows, such as when moving files between file browser windows or when moving text between documents. With little practice, our participants were up to 47% faster with Depth-Drag than with a regular mouse. Our research on the fundamental desktop interactions for touch-based mouse input provides the necessary foundation to support and push development further in this direction.

REFERENCES

Ahlström, D., Großmann, J., Tak, S., and Hitz, M. (2009). Exploring new window manipulation techniques. In *OzCHI '09*, pages 177–183. ACM Press.

Ahlström, D., Hitz, M., and Leitner, G. (2006). An evaluation of sticky and force enhanced targets in multi-target situations. In *NordiCHI'06*, pages 58–67. ACM.

Ahmad, B. I., Langdon, P. M., Bunch, P., and Godsill, S. J. (2014). Probabilistic intentionality prediction for target selection based on partial cursor tracks. In *UAHCI 2014*, pages 427–438. Springer.

Ahn, S., Lee, J., Park, K., and Lee, G. (2018). Evaluation of edge-based interaction on a square smartwatch. *Int. J. Hum. Comput. Stud.*, 109:68–78.

Apple Inc. (2022). Magic mouse-white multi-touch surface.

Bartindale, T., Harrison, C., Olivier, P., and Hudson, S. E. (2010). Surfacemouse: Supplementing multi-touch interaction with a virtual mouse. In *TEI '11*, page 293–296. ACM Press.

Benko, H., Izadi, S., Wilson, A. D., Cao, X., Rosenfeld, D., and Hinckley, K. (2010). Design and evaluation of interaction models for multi-touch mice. In *Graph. Interface*, pages 253–260. Canadian Human-Computer Communications Society.

Casiez, G., Vogel, D., Balakrishnan, R., and Cockburn, A. (2008). The impact of control-display gain on user performance in pointing tasks. *Hum.-Comput. Interact.*, 23(3):215–250.

Conroy, E., Toth, A. J., and Campbell, M. J. (2022). The effect of computer mouse mass on target acquisition performance among action video gamers. *Appl. Ergon.*, 99:103637.

Franz, J., Menin, A., and Nedel, L. (2016). Lossless multitasking: Using 3d gestures embedded in mouse devices. In *IEEE SVAR*, pages 109–116. IEEE.

Froehlich, B., Hochstrate, J., Skuk, V., and Huckauf, A. (2006). The globefish and the globemouse: Two new six degree of freedom input devices for graphics applications. In *CHI '06*, pages 191–199. ACM Press.

Grossman, T. and Balakrishnan, R. (2005). The bubble cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. In *CHI '05*, page 281–290. ACM Press.

Hart, S. G. and Staveland, L. E. (1988). Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Adv. Psychol.*, volume 52, pages 139–183. Elsevier.

Hinckley, K. and Sinclair, M. (1999). Touch-sensing input devices. In *CHI '99*, pages 223–230. ACM Press.

Hinckley, K., Sinclair, M., Hanson, E., Szeliski, R., and Conway, M. (1999). The videomouse: A camera-based multi-degree-of-freedom input device. In *UIST '99*, pages 103–112. ACM Press.

Hourcade, J. P., Crowther, M., and Hunt, L. (2007). Does mouse size affect study and evaluation results? a study comparing preschool children's performance with small and regular-sized mice. In *IDC '07*, pages 109–116. ACM Press.

Hutchings, D. R., Smith, G., Meyers, B., Czerwinski, M., and Robertson, G. (2004). Display space usage and window management operation comparisons between single monitor and multiple monitor users. In *AVI '04*, pages 32–39. ACM Press.

Kim, S., Lee, B., Van Gemert, T., and Oulasvirta, A. (2020). Optimal sensor position for a computer mouse. In *CHI '20*, pages 1–13. ACM Press.

Martin, B. and Raisamo, R. (2004). Trackmouse: A new solution for 2+2d interactions. In *NordiCHI '04*, pages 89–92. ACM Press.

McGuffin, M. and Balakrishnan, R. (2002). Acquisition of expanding targets. In *CHI '02*, pages 57–64. ACM.

Microsoft Corporation (2022). Microsoft arc mouse.

Mistry, P. and Maes, P. (2011). Mouseless: A computer mouse as small as invisible. In *CHI EA '11*, pages 1099–1104. ACM Press.

Ortega, M. and Nigay, L. (2009). Airmouse: Finger gesture for 2d and 3d interaction. In *Human-Computer Interaction – INTERACT 2009*, pages 214–227. Springer.

Perelman, G., Serrano, M., Raynal, M., Picard, C., Derras, M., and Dubois, E. (2015). The roly-poly mouse: Designing a rolling input device unifying 2d and 3d interaction. In *CHI '15*, pages 327–336. ACM Press.

Peters, M. and Ivanoff, J. (1999). Performance asymmetries in computer mouse control of right-handers, and left-handers with left- and right-handed mouse experience. *J. Motor Behav.*, 31(1):86–94.

Roth, V. and Turner, T. (2009). Bezel swipe: Conflict-free scrolling and multiple selection on mobile touch screen devices. In *CHI '09*, pages 1523–1526. ACM.

Saidi, H., Serrano, M., Irani, P., and Dubois, E. (2017). Tdome: A touch-enabled 6dof interactive device for multi-display environments. In *CHI '17*, pages 5892–5904. ACM Press.

Sato, K., Imada, S., Mazume, S., and Nakashima, M. (2020). A mechanism of window switching prediction based on user operation history to facilitate multitasking. In Barolli, L., Nishino, H., Enokido, T., and Takizawa, M., editors, *Adv. Networked-Based Inf. Syst.*, pages 154–166. Springer.

Serrano, M., Lecolinet, E., and Guiard, Y. (2013). Bezel-tap gestures: Quick activation of commands from sleep mode on tablets. In *CHI '13*, pages 3027–3036. ACM.

Shi, K., Subramanian, S., and Irani, P. (2009). Pressuremove: Pressure input with mouse movement. In *INTERACT 2009*, pages 25–39. Springer.

Smith, G., Baudisch, P., Robertson, G., Czerwinski, M., Meyers, B., Robbins, D., and Andrews, D. (2003). GroupBar: The TaskBar evolve. In *OzCHI '03*, pages 34–43. ACM Press.

Son, S., Jung, J., Ham, A., and Lee, G. (2023). Touchwheel: Enabling flick-and-stop interaction on the mouse wheel. *Int. J. Hum.-Comput. Inter.*, 0(0):1–13.

Tak, S., Cockburn, A., Humm, K., Ahlström, D., Gutwin, C., and Scarr, J. (2009). Improving window switching interfaces. In *INTERACT 2009*, pages 187–200. Springer.

Tak, S., Scarr, J., Gutwin, C., and Cockburn, A. (2011). Supporting window switching with spatially consistent thumbnail zones: Design and evaluation. In *INTERACT 2011*, pages 331–347. Springer.

Tashman, C. (2006). Windowscape: a task oriented window manager. In *UIST '06*, pages 77–80. ACM Press.

Tashman, C. and Edwards, W. K. (2012). Windowscape: Lessons learned from a task-centric window manager. *ACM Trans. Comput.-Hum. Interact.*, 19(1):8:1–8:33.

Taylor, S., Keskin, C., Hilliges, O., Izadi, S., and Helmes, J. (2014). Type–hover–swipe in 96 bytes: A motion sensing mechanical keyboard. In *CHI '14*, pages 1695–1704. ACM Press.

Ultraleap (2013). Leap motion controller.

Villar, N., Izadi, S., Rosenfeld, D., Benko, H., Helmes, J., Westhues, J., Hodges, S., Ofek, E., Butler, A., Cao, X., et al. (2009). Mouse 2.0: Multi-touch meets the mouse. In *UIST '09*, pages 33–42. ACM Press.

Warr, A., Chi, E. H., Harris, H., Kuscher, A., Chen, J., Flack, R., and Jitkoff, N. (2016). Window shopping: A study of desktop window switching. In *CHI '16*, pages 3335–3338. ACM Press.

Xu, Q. and Casiez, G. (2010). Push-and-pull switching: Window switching based on window overlapping. In *CHI '10*, pages 1335–1338. ACM Press.

Yang, X.-D., Mak, E., McCallum, D., Irani, P., Cao, X., and Izadi, S. (2010). Lensmouse: Augmenting the mouse with an interactive touch display. In *CHI '10*, pages 2431–2440. ACM Press.

Zeidler, C., Lutteroth, C., and Weber, G. (2013). An evaluation of stacking and tiling features within the traditional desktop metaphor. In Kotzé, P., Marsden, G., Lindgaard, G., Wesson, J., and Winckler, M., editors, *INTERACT 2013*, pages 702–719. Springer.