Cross-Device Shortcuts: Seamless Attention-guided Content Transfer via Opportunistic Deep Links between Apps and Devices

Marilou Beyeler*
Department of Computer Science
ETH Zürich, Switzerland

Yi Fei Cheng*
Department of Computer Science
ETH Zürich, Switzerland

Christian Holz
Department of Computer Science
ETH Zürich, Switzerland

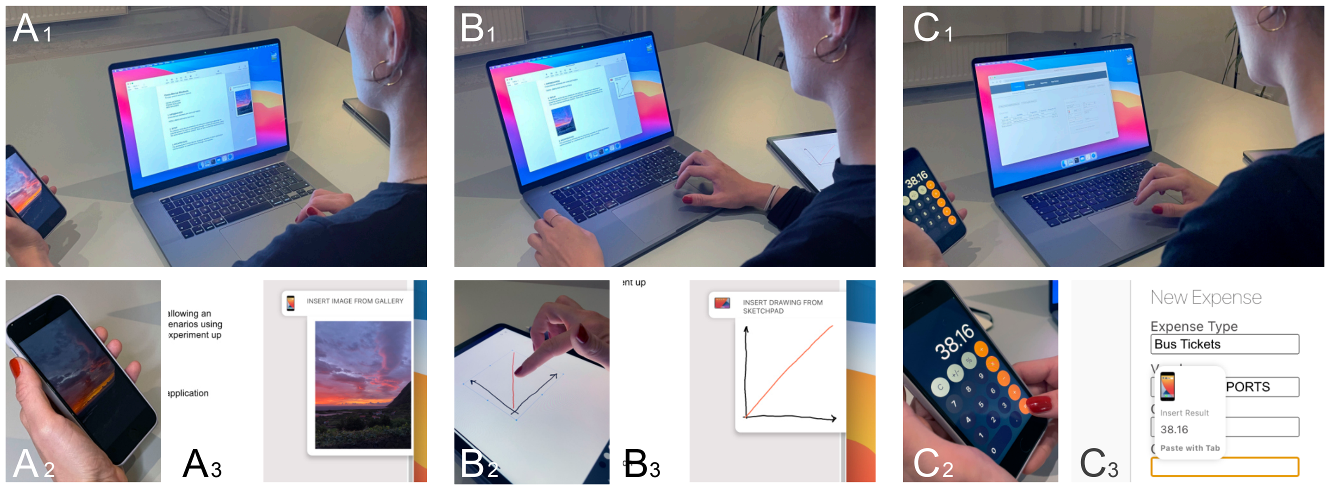

Figure 1: *Cross-device shortcuts* directly link content from an app running on one device into an app on another—*content-aware*, manifesting following the user's *attention switches*, and presented via *feed forward*. (A) This user selects a picture on a phone and then attends to her laptop, creating a cross-device shortcut that is revealed through an in-app preview and afford drag&drop. (B) Cross-device shortcuts work across phones, laptops, and tablets. (C) Feed-forward previews appear at the UI-element level, in-app level, or operating-system level depending on content type and the receiving app's capability for accepting it.

ABSTRACT

Although users increasingly spread their activities across multiple devices—even to accomplish a single task—information transfer between apps on separate devices still incurs non-negligible effort and time overhead. These interaction flows would considerably benefit from more seamless cross-device interaction that *directly* connects the information flow between the involved apps across devices. In this paper, we propose *cross-device shortcuts*, an interaction technique that enables direct and discoverable content exchange between apps on different devices. When users switch their *attention* between multiple engaged devices as part of a workflow, our system establishes a cross-device shortcut—a deep link between apps on separate devices that presents itself through *feed-forward previews*, inviting and facilitating quick content transfer. We explore the use of this technique in four scenarios spanning multiple devices and applications, and highlight the potential, limitations, and challenges of its design with a preliminary evaluation.

*These two authors contributed equally to this work.

CCS CONCEPTS

• **Human-centered computing → Interaction techniques.**

KEYWORDS

Cross-device interaction, content transfer, discoverability

ACM Reference Format:
Marilou Beyeler, Yi Fei Cheng, and Christian Holz. 2023. Cross-Device Shortcuts: Seamless Attention-guided Content Transfer via Opportunistic Deep Links between Apps and Devices. In *INTERNATIONAL CONFERENCE ON MULTIMODAL INTERACTION (ICMI '23), October 09–13, 2023, Paris, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3577190.3614145

1 INTRODUCTION

End-users increasingly involve multiple devices as part of their workflows [4, 46]. Such spontaneous device setups provide advantageous multi-screen affordances [7, 16, 29, 42] and enable users to productively distribute their sub-tasks across devices [45]. While authoring a document, for instance, users may supplement their use of a computer with companion devices, such as a tablet for sketching or a phone for inserting photographs.

Researchers and product designers have recognized the need to support such impromptu uses of devices, and have accordingly implemented a considerable number of frameworks, interaction techniques, and systems for creating integrated device ecologies [5, 30]. However, despite significant research efforts and commercial systems for cross-device interactions (e.g., Apple Continuity [21], Samsung Flow [38]), fluid information transfer across devices remains a persistent challenge [46]. Many factors contribute to this barrier, but one of the most notable is users' difficulties in *discovering* cross-device interaction possibilities [1, 4].

Consider the scenario of supplementing the document authoring process with content from companion devices. Using current infrastructure for cross-device interactions (e.g., Apple Continuity [21], Universal Clipboard [20]), users must *explicitly* initiate all content transfer by selecting it on their companion device and pasting it on their primary device. While straightforward, current copy & paste approaches do not opportunistically exploit natural user behaviors, which may reduce the viscosity of transfer [23, 39]. Further, they fail to provide users with an awareness of the to-be-transferred content—they reveal this action only *after the fact*. The lack of such mechanisms contributes to the presence of a *Gulf of Execution* (i.e., the gap between user intention and allowable system actions) [19, 33, 34], thereby causing undesirable situations (e.g., transfer of incorrect contents) and hindering interaction.

To address these challenges, we introduce *cross-device shortcuts*, an interaction technique that facilitates direct and discoverable content exchange across devices and the apps running on them. To support fluid information transfer, our technique is informed by how users naturally interact across devices. We enable content transfers to be initiated on the basis of both intentional and implicit interaction. Our technique relays content between devices by inferring the user's attention, which we approximate using gaze tracking from the device's camera feed. To increase the discoverability of context exchange opportunities, cross-device shortcuts present feed-forward previews of potential deep links across applications on different devices at multiple UI levels. In contrast to prior approaches that either showed no feed-forward or presented it only on the operating system level, our multi-level feed-forward previews enable deeper linkages and present information where it is most relevant. For this purpose, our system tailors feed-forward previews depending on content type.

We present a series of apps that leverage cross-device shortcuts and discuss insights from our interactive walk-through study with researchers and multi-device users, highlighting the notable trade-off between discoverability and system intrusiveness in cross-device interaction. We also discuss the challenges in designing opportunistic content transfer for understandability and learnability.

Contributions

(1) A cross-device interaction technique that combines attention-driven content transfer, content-type awareness, and multi-level feed-forward notifications to establish discoverable deep links between apps across devices.
(2) A series of applications that demonstrate cross-device shortcuts in diverse everyday usage scenarios.
(3) Insights from expert users and cross-device researchers through an interactive walk-through study.

2 BACKGROUND AND RELATED WORK

Cross-device shortcuts are situated in the landscape of existing cross-device interaction techniques and technologies. For a comprehensive overview, we refer readers to Brudy et al.'s cross-device taxonomy [4]. Our primary design goal was to facilitate two of the multi-device usage patterns highlighted by Yuan et al. [46]: *integrating* devices that are suited for the task and *partitioning* tasks onto different devices. Our technique aims to reduce the viscosity of cross-device information transfer, which has been a persistent challenge [23, 39]. Our design was informed by four key features of prior transfer techniques: method of initiation, method of transfer, content-type awareness, and feed-forward.

2.1 Explicit or implicit initiation

One distinguishing dimension between existing cross-device techniques is whether content transfer is initiated explicitly or implicitly. Explicit approaches, such as Pick-and-Drop [36] or Conduits [7], require explicit user action. Implicit initiations, such as in Conductor [16] and Gradual Engagement [28], are driven by observed user behaviors. While the former class of approaches are more straightforward because they are always user initiated, they are may be less directly integrated into natural user behaviors. Implicit approaches are more integrated, but require an accurate model of user interaction, which is not always available. Our technique supports both methods of initiating content transfer.

2.2 Method of content transfer

Content transfer techniques differ in how source and target devices are linked [8]. Several common approaches include extending the concept of single device clipboards (e.g. copy & paste) for multiple devices (e.g., Apple Universal Clipboard [20]), mediating the transfer with the user's hands (e.g., Pick-and-Drop [36]), and leveraging spatial tracking (e.g., GradualEngagement [28], Tracko [22], PolarTrack [35]) or temporal proximity [5, 17]. Several prior techniques have used attention tracking, through which source and target devices can be linked and interaction can be integrated into everyday workflows [26]. GazeConduits supports collaborative interaction through mobile gaze estimation [44]. SMAC supports fluid interaction across devices using a model of user attention [26]. Gluey 'glued' content onto the user's visual field for transferring it across devices using a see-through display [40]. The design of cross-device shortcuts builds on these approaches with content type-aware feed-forward to support the discovery of interactions.

2.3 Content-type awareness

Content-type awareness refers to a system's ability to make contextually appropriate decisions based on the characteristics of content that users interact with as part of an ongoing task. Citrine is an early example that intelligently transforms the contents of copy&paste operations based on what it consists of and the output app [41]. Conductor also supports content type-aware cross-device interaction [16], using a content-adaptive 'cue' to transfer information. These projects showed that content-type awareness is valuable for enabling smooth cross-device content exchange. Cross-device shortcuts build on these, relying on content-type awareness to mediate the display of content transfer opportunities.

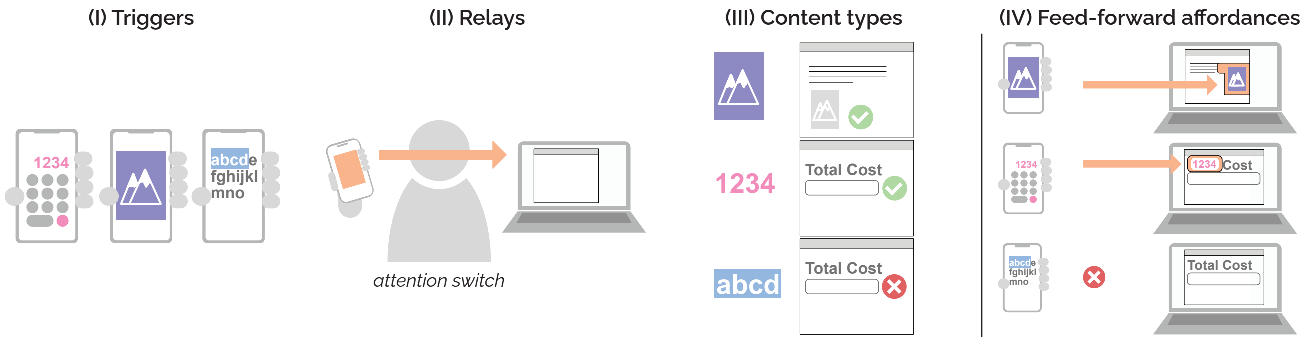

Figure 2: The four main interaction stages of cross-device shortcuts alongside the logic and flow of content.

2.4 Feed-forward

While the significance of feedback in interaction design is well established among interaction designers [43], prior work has shown that feed-forwards are equally important [12, 32, 37]. Whereas feedback informs the user of the result of an action *after the fact*, feed-forward invites action by conveying the action's result before execution [11, 12, 43]. By showing users a preview of their actions' results, well-designed feed-forward may help bridge Norman's Gulf of Execution [19, 33, 34]. Feed-forward can increase the discoverability and learnability of interaction techniques, like marking menu usage [24], force-based text-selection [13], and pen gestures [2]. In cross-device contexts, feed-forward could also support functionality discovery [1], especially for transferring content. For example, Conductor [16] displayed options for duplicating states when users switched between devices to support access to this feature.

Whereas copy&paste across devices (e.g., Windows Cloud Clipboard [31], Apple Universal Clipboard [20]) presents the user with a *fait accompli* by inserting content, feed-forward provides a discoverable presence *prior* to pasting. Prior usage of feed-forward for cross-device information transfer is limited, and it is restricted to notifications on an operating system level. Apps such as Clippy [9] and Pasteasy [27] support feed-forward in one direction (e.g., PC-to-phone). In contrast to prior work, cross-device shortcuts enable deeper content linkages across devices and present opportunities for information transfer at multiple interface levels (i.e., UI level, in-app level, operating system level). We demonstrate this functionality across a wider set of devices and compliment it with content-type awareness and attention detection to reduce transfer viscosity.

2.5 Summary

Overall, cross-device shortcuts implement a novel combination of attention-driven content transfer, content-type awareness, and multi-level feed-forward to support *deeper* and more discoverable linkages between cross-device content. Our approach is designed to build on existing interaction patterns while enabling users to specify their intentions in a straightforward manner. We leverage known techniques in modeling user attention to link source and target devices, an approach that has been validated to support more fluid interactions across devices [26]. We distinguish our work from prior research in how we complement these mechanisms with content-aware multi-level feed-forward previews. With feed-forwards, our approach aims to enable users to maintain a greater sense of situational awareness of the information flow. We argue that it is beneficial to present feed-forwards hierarchically (i.e., at the UI-element, in-app, or operating system level) to provide a preview of the transfer contents *where* it is most relevant to the user's interactions. Likewise, mediating the presentation of feed-forwards based on the contextual relevance of the transfer contents (i.e., content-type awareness) assists the user to providing them with direct access to *what* they may want to transfer.

3 CROSS-DEVICE SHORTCUTS

Cross-device shortcuts support quick and convenient content transfer between elements inside apps on different devices. Shortcuts manifest based on the user's presence, attention, and the semantic context of the ongoing task. They are announced through feed-forward for discoverability and fluid interaction. Here, we present the design of our interaction technique.

3.1 The 4 stages of cross-device shortcuts

Interacting betweem multiple devices using cross-device shortcuts comprises four stages as outlined in Figure 2.

I) Triggers: Issuing events that select or create an information entity initiate all cross-device shortcuts. These events are registered on a source device following the user's input.

II) Relays: The user implicitly links a source device with a target device by switching their attention from the former to the latter while engaging with both in the process of a task.

III) Content types: The media type of the information entity on the source device along with the target device's semantic context—the capability of accepting this content type—determine whether a shortcut will be established.

IV) Feed-forward previews: Notifications on the target device reveal an established cross-device shortcut. They appear at one of three hierarchy levels: UI-element level inside an app, in-app level, or operating-system level. From there, users can directly insert the content or, alternatively, store it on the device for later use.

To describe these four stages in detail and give design rationales, we consult the following scenario: a user is writing an email on her laptop. To insert a recently taken image, she picks up her phone, opens the photo gallery, and flicks through individual photos. Once she has found a suitable photo (I), she moves her attention back to the laptop (II), which still shows the email editor (III). A notification appears inside the email (IV), showing a preview of the photo, which she simply drags into the email and continues writing.

I) Triggers

Cross-device shortcuts start with triggers that follow input events from the user. Triggers specify the content designated to enter a shortcut for consumption on the receiving end. In our case, triggers can be either explicit or implicit. An *explicit trigger* is an action that is not followed by a follow-up input event, hence carrying meaning and indicating intent. Examples include highlighting text and selecting a file in the file system or an attachment inside an email. When a user creates new content following an input event, the result represents an *implicit trigger.* An implicit triggers thus follows the completion of an (inter)action on the device. Examples include opening an item in a gallery (i.e., resulting in full display), pressing '=' in Calculator, and hitting 'Done' or 'Apply' in an editor (i.e., concluding an operation that results in a new medium).

II) Relays

Within a short period following a trigger event, a relay event can establish a cross-device shortcut to a target device or app. Relays are user actions that involve multiple devices. Through relays, the user establishes a cross-device shortcut. Relay events succeed when a user turns their attention from one device to another while remaining engaged with both. Events that indicate continued engagement include keeping the phone or tablet in hand or placing it down close by. In contrast, pushing a phone or tablet aside, turning it off, placing it face-down, or letting it idle for too long all represent indications of discontinued engagement. In these cases, we abort the relay. Relays expire shortly after a trigger if none of the actions above are detected. Relays also become obsolete if a novel trigger supersedes a prior one or the user undoes a previous trigger (e.g., by deselecting text).

III) Content types

Once content enters a cross-device shortcut following a relay, the target device determines if the shortcut should manifest. Here, content types (e.g., text, numbers, images, vectors, files) become relevant. When content enters a shortcut, success depends on whether the receiving app (i.e., client app in focus) accepts its media type. For example, a text field accepts all types of text, whereas a phone number field only accepts digits. If the target app does not accept the text (e.g., in the case of an audio editor), the shortcut fails. In contrast, shortcuts always succeed for file-based content.

IV) Feed-forward previews

If the target device accepts the cross-device shortcut, it displays a feed-forward preview through a notification. The notification announces an established shortcut and previews its contents, inviting users to continue their interaction flow on the target device. The shortcut notification appears on one of three hierarchy levels depending on content type and the semantics of the current task: UI-element level, in-app level, or (global) operating-system level.

UI-element feed-forward. These affordances show the source device's icon and content preview right by the focused UI element (Figure 3 left). Pressing 'tab' or clicking the icon pastes the content. A notification can be discarded by pressing the 'escape'. If the user focuses on another UI field, the icon follows the selection to the now-focused element if it accepts the shortcut's content type.

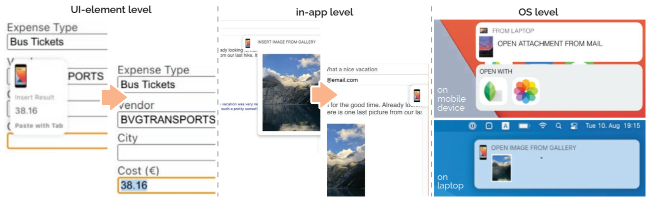

Figure 3: Successfully established cross-device shortcuts present one of three feed-forward previews.

Otherwise, the notification icon migrates to an in-app notification, so it remains accessible for later pasting.

In-app feed-forward. These affordances appear for text or file-based shortcut content if the app accepts the content type. An in-app notification appears as a small panel sliding in from the right app border, previewing the shortcut contents alongside the source's device icon (Figure 3 center). The user can then drag it into the app to place the content. To avoid obstructing app content, preview tabs rescind to an icon after a short period. Clicking the rescinded icon then opens the preview again. If the target app does not support the shortcut content type, the target device propagates the cross-device shortcut up the hierarchy to the operating-system level.

Operating system shortcut feed-forward. These affordances represent a general-purpose fallback option. On desktop devices, they materialize for file-based shortcut contents or content whose type does not match the app in focus. Their appearance on the desktop resembles that of regular system notifications, with the addition of a small preview of the source device and content (Figure 3 right). From there, the user can either drag a notifications into an app that accepts the content type or onto the desktop to create a file. Alternatively, clicking the notification opens the file with the system viewer. On mobile devices, cross-device shortcuts produce global notifications, because no notion of a desktop that stores files exists. Tapping the notification opens a menu with options of list of suitable apps (Figure 3 right).

3.2 Prototype Implementation

As a proof-of-concept of our interaction techniques, we prototyped cross-device shortcuts as a live system to allow end users to experience the cross-system behavior enabled by our techniques. For this, we implemented a client-server architecture that comprises a content relay backend and a service layer that runs on all clients. These two handle all the communication and message exchange to seamlessly facilitate our technique. Additionally, the service layer running on the clients sources all input from users, including direct interactions as well as attention and processes actions.

To enable our scenarios, we created prototype apps that resemble the full-fledged counterparts running on macOS and iOS. Our desktop front-end apps include Mail, Pages (a rich-text editor), and Terminal (a console app). All support similar interaction, behavior, and part of the functionality of their desktop equivalents. For smartphones and tablets running iOS, we developed apps to resemble the Gallery (to browse photos), Camera (to take pictures), Mail, Notes (a sketching tool), Preview (an annotation app), and Snapseed (a photo editor). We also developed a basic expense reporting app to study the use of a more comprehensive interaction session. The app

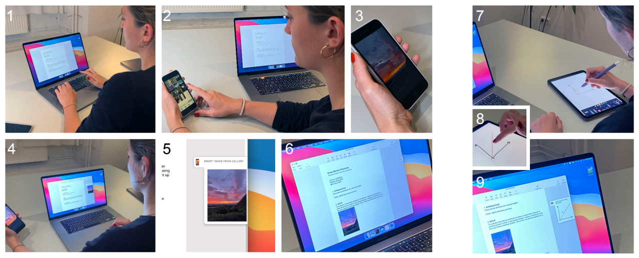

Figure 4: Shortcuts allow inserting (1–2) content from one device into (3–4) an app on another. (5) A feed-forward notification previews the content, which can be dragged into (6) this document editor. (7–9) A similar workflow with sketches.

spans multiple types of content, snippets and files, various content fields and steps to complete a full report. As opposed to deploying in existing applications, which would require inconveniently deep access to application functionalities, we developed new apps to showcase and evaluate our interaction techniques such that we could focus our investigation on the interaction level.

Technical implementation. We implemented the service layer as well as all front-end apps in React.js. The backend server was implemented in Node.js. All client apps and systems communicate with the backend through WebSockets with serialized JSON structures.

The service layer on clients implements attention tracking by continuously capturing the front-facing camera feed in a concurrent thread. Our implementation processes these videos through Google's Vision API [10] to extract faces and estimate the user's gaze direction. Since each instance of our service layer is aware of the type of device it is running on as well as its physical dimensions, we define a small region around the device that is informed by the device's orientation, which our implementation derives from the built-in inertial sensor. If the gaze direction returned by the Vision API falls within this region, a client infers that the user's attention is currently directed at this device. This conclusion is transferred to the server, which relays it to all clients. If two devices share line of sight, the user's attention is inferred to be focused on the device where their head appeared largest.

To detect physical activity and interaction beyond direct touch input, our service layer concurrently monitors the built-in inertial sensors on all mobile devices. Our clients process the continuous stream of accelerations for events such as holding the device, placing it down, picking it up, pushing it away through a simple analysis of the orientation vector (held vs. flat on the table), and basic thresholding and smoothing of acceleration magnitude over time to distinguish a stationary device from one that is being used or held in hand. From the reported accelerations, we also coarsely approximate the direction of motion for each device (i.e., when pushing aside) and the duration of this motion.

Each client notifies the backend about trigger candidates (i.e., selections and results including media types) and relay events (i.e., face turning away, face turning towards, and physical activities). For the period during which relays can be established, we chose 15 s within the user needs to demonstrate activity (e.g., an attention switch, engaging with the device). Upon detecting a valid relay,

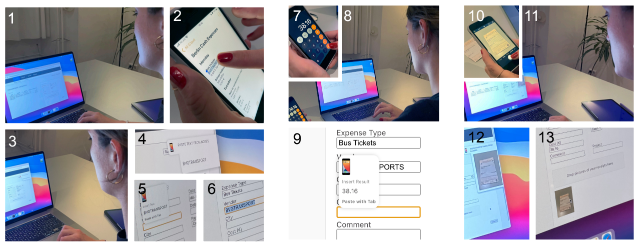

Figure 5: Shortcuts are content-type aware. To fill this expense report, (1–2) the user selects text, (3) turns to her laptop, which (4) creates a shortcut and, (5) due to matching content types, shows a UI-element notification. (7–9) A shortcut results from hitting '=' in Calculator. (10–13) This freshly taken picture enters a shortcut but does not match the form fields' content type, therefore creating in-app feed-forward.

the server determines the target of the cross-device shortcut and inquires with the corresponding client. The client validates the content type and decides on the feed-forward preview to display.

4 APPLICATIONS

We now describe several applications and how they leverage cross-device shortcuts to enable seamless interaction across devices.

Media integration into a rich-text editor. A user is creating a report using a rich-text editor on her laptop (Figure 4). She browses images on her phone to add to her report. After selecting an image (*trigger*, Figure 4.3), she switches her attention back to the text editor on her laptop (*relay*, Figure 4.4). A cross-device shortcut is established between her phone and laptop and a feed-forward preview of the selected picture appears inside an in-app notification (Figure 4.5). She then drags and places the picture into the text and resumes writing. Later, she scrolls through the pictures on her phone again. However, she decides against adding another image and instead turns off the phone. As she switches her attention back to the report, no shortcut emerges and no notification appears on her desktop since her phone is no longer engaged. Figure 4.8 shows a similar process for adding a sketch to the report.

Online expense report. The user now fills out an expense report using a web app. She picks up her phone and searches her emails for the reference details to enter (Figure 5.2), selects the text, and switches her attention to the computer. The selected text initially appears as a feed-forward in-app notification (Figure 5.4). When she clicks the form field 'Costs', the notification rescinds, because the field only accepts numbers (*content-type awareness*). She then moves the cursor to the 'Vendor' field, which causes the notification to morph into UI-element feed-forward (Figure 5.5). She presses 'tab', which fills the shortcut content into the field (Figure 5.6). Afterwards, she moves back to the 'Costs' field, where the expenses app requires her to enter the total spent on public transport. She calculates her expenses on her phone Calculator app. The 'equals' button acts as a trigger in this case. When she turns back to the laptop, a shortcut is shown as a UI-level feed-forward on top of the selected 'Costs' field (Figure 5.9). Finally, she scans a receipt using her phone's camera app. When she switches her attention back to the form on her laptop, an in-app feed-forward preview shows the photo, which she then drags into the form (Figure 5.12–13).

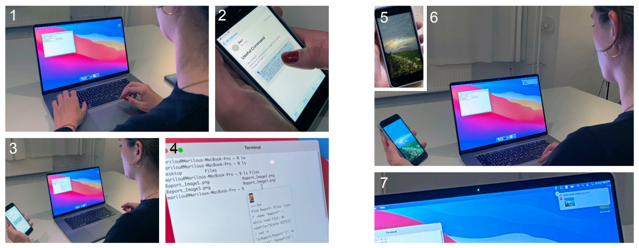
Figure 6: (1–4) A cross-device shortcut from text on a phone to the laptop's terminal results in a notification by the cursor. (5–6) When an image is selected instead, the shortcut is still established, but produces an operating-system notification.

Terminal commands. Using the laptop's terminal, the user wants to execute a complex command that she has stored on her phone. She selects it and turns her attention back to the terminal. This creates a cross-device shortcut previewing the selected text above the cursor (Figure 6.4). She presses 'tab' to insert the command and then executes it. Still on her phone, she selects a picture from the gallery and turns her attention back to her laptop, where the terminal is still active. As the terminal does not accept images, no in-app notification appears. Instead, a global desktop feed-forward notification appears (Figure 6.7). She can then drag the image from the notification to the desktop to store it for later use.

Document annotation. Here, the user is annotating an email on her laptop, but using inking on her tablet. She first selects the attachment and then turns to the tablet. This shows an OS-level notification with a preview of the document (Figure 7.2). She taps the notification, selects the annotation app (Figure 7.3), which opens the document. After annotating, she taps 'Done' and switches her attention back to her laptop. This creates a feed-forward notification on the desktop. She then drags the preview into her email response (Figure 7.6). As shown in Figure 7.7–10), the same steps allow her to edit a photo from her laptop, using custom tools on her phone.

5 PRELIMINARY WALK-THROUGH STUDY

To gather preliminary feedback on cross-device shortcuts, we conducted a remote interactive walk-through study with 8 participants (3 female, 5 male, ages 25–43) for 60–90 minutes each. As an established evaluation technique [25], we saw a primary value in conducting a walk-through demonstration over a usability study to explore how people may use our technique, while still gaining valuable feedback on the utility of its current features. Prior work has further suggested that the early stages of interaction design benefit from exploratory and qualitative evaluations, especially with experts [6, 15, 18]. As such, in our study, we focused on eliciting qualitative feedback from experts and power users.

5.1 Participants

We recruited four international expert researchers in the multi-device space with nine or more years of experience designing, developing, and evaluating cross-device applications and techniques (E1–E4). All experts had published 8 or more papers on cross-device interaction techniques or frameworks at ACM ICMI, UIST, or CHI.

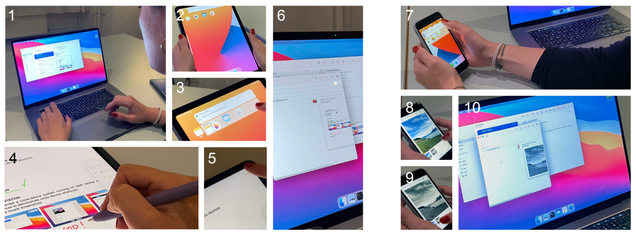
Figure 7: Round-trip crossing to (1–4) annotate an email attachment on the laptop by inking on the tablet and (5–6) transferring it back. (7) A similar procedure to (8–9) apply a filter in a photo app and (10) transfer the result back.

The other four participants were power users of multi-device ecologies in their daily work (P1–P4), working with and across three or more devices, often simultaneously, to accomplish the tasks for their jobs. All participants owned and used a phone and a tablet on at least a daily basis in addition to their computer.

5.2 Procedure

The experimenter demonstrated the four scenarios described in Section 4, showing the features of the system, and explaining the system behavior after each interaction. Participants could operate with the example applications through the experimenter as a proxy in real-time. The experimenter also interactively customized the demonstrations based on participant requests to revisit steps or show interactions with different source content or target apps. Participants were repeatedly encouraged to interrupt at any time for feedback or questions. Discussions emerged spontaneously, informed by the experimenter's initial open-ended questions to guide the conversation. All studies were video recorded.

5.3 Results

Our participants were generally enthusiastic about cross-device shortcuts and related the technique to personal scenarios; however, they were also concerned about the intrusiveness of feed-forwards and the risks of driving system behaviors with implicit interactions.

5.3.1 Reflections on demonstrated scenarios. The diversity of demonstrated scenarios allowed each participant to find an aspect to relate to. For example, P2 mentioned the usefulness of being able to seamlessly transition between using a computer keyboard to take notes and a tablet for sketching. Reflecting on the presented usage of a calculator, P4 remarked: *"I work with the calculator a lot and often make typos when typing the result in my document. This [cross-device shortcut] would make it super fast and remove mistakes."*

All but two participants related to the (cross-device) effort of filing the expense report, where they had themselves involved multiple types of content from different devices. E2 reported using Google Keep [14] to transfer expense details and saw cross-device shortcuts as a more direct integration. P2 and P3 said that they had found synchronized cloud clipboards to work well for text transfers in such scenarios, but not images. Commenting on shortcuts, P3 said *"It is practical that you can transfer all types of contents all through the same steps."*

Lastly, all participants pointed out back-and-forth transfer as a frequent use-case that they find little support for in current tools.

P2, a teacher, said *"[when grading students' homework] ... I'd like to annotate the file but don't want it permanently stored on my tablet. Transferring and organizing [files] is a lot of effort."*

5.3.2 Valued aspects. Participants most significantly appreciated how cross-device shortcuts as a whole often emerged just in time ($N = 8$). E1 and P1 called it explicit enough to support specific operations, while building on top of smooth and natural behavior. While the experimenter did not explicitly use our system terms 'trigger' or 'relay' during the walk-through sessions, we use them below to group participants' comments:

Triggers. All participants found valuable aspects *explicit* and *implicit triggers.* On explicit triggers, P3 said *"I like that I can just select and I don't have to press [anything to] copy."* On implicit triggers, P4 said that *"It's nice that I don't have to select and copy it. [It feels] more like a real calculator."*

Relays. Participants ($N = 4$) particularly responded to shortcut relays, which combine device engagement and attention switches. P1 commented that relaying information between devices through a head turn *"feels natural"*.

Content-type awareness. Participants ($N = 6$) were positive about the content-type awareness of cross-device shortcuts. P3 compared it to auto suggestions, saying *"it's like that feature that fills out your email address in the right field through a single click. But here I can control over the value with my phone."*

Feed-forwards. 5 participants explicitly called out feed-forward as a core advantage over existing copy & paste. E1 remarked that *"there is definitely a lot missing in current cross-device clipboards. [...] visual feedback and clarity about what is done when and when it is available goes a long way."* All participants expressed a preference for notifications on different hierarchy levels. All but one called them *'subtle'* or *'non-intrusive'*. Multiple participants described its advantage to avoid wrong insertions in the event of false-positives ($N = 3$). Several participants appreciated how in-app notifications appeared *right where needed* ($N = 3$).

5.3.3 Intrusiveness. While participants saw value in having cross-device content transfers initiated through implicit means and previewed through feed-forwards, they also worried about the potential intrusiveness of this approach ($N = 4$). Several participants feared that false-positive activation of cross-device shortcuts could lead to unwanted feed-forward previews. Participants ($N = 3$) additionally considered the technique's use of notifications as potentially cumbersome: *"I often have notifications turned off because I sometimes get annoyed. Maybe there should be a fast way to see the shortcuts only when I want"* (P1). P3 voiced skepticism about the utility of having the feed-forward preview follow the cursor across fields. As an alternative, E3 recommended switching to static UI notifications that only appear next to the respective field.

5.3.4 Unexpected device behaviors. While participants appreciated how cross-device shortcuts could be initiated both implicitly and explicitly, they also voiced concerns about how the inclusion of both may lead to unexpected confusing device behaviors ($N = 4$). E2 reported, *"With the calculator, I wouldn't really know, is it my result transferred when I switch my attention? Maybe it is my input. It only becomes clear when you see the notification pops up."* As a result, cross-device shortcuts may require a learning phase to understand and trust the system. For example, E3 commented: *"Holding your phone is a quite natural, implicit interaction. Users might not notice that this is an indicator for the system. That could be frustrating when it does not work."* Two of the expert users noted that despite how cross-device shortcuts were generally *'intuitive'* and *'activate[d] at the right moment'*, system decisions may be difficult to understand early on. P2 summarized, *"in the beginning, you might not understand what is happening and how to initiate it."* P3 suggested potentially including feedback explaining *why* the system had activated (e.g., saying *"attention shift detected"*).

Closely related to the aforementioned concerns, half the participants further remarked that having sometimes ambiguous and implicit transfers in such a seamlessly integrated ecosystem of devices could pose a significant risk to their privacy. Some mentioned ($N = 3$) that they have personal photos on their phones that they feared may accidentally pop up on the screen of their work laptop. P2 stated *"If I'm in a video call sharing my screen I'd be scared that private picture could pop up."* P1 suggested *"Maybe the notification could stay closed and open only manually in those cases."*

6 DISCUSSION

Overall, our walk-through demonstration study with expert researchers and power users of cross-device workflows revealed several insights about the promises and challenges of our approach.

6.1 Understandability & consistency

Participants appreciated how cross-device shortcuts enabled more seamless information transfer; however, their comments also showed that this has potential downsides, especially for novice users.

To *relay* cross-device shortcuts, our system requires an attention shift during continued engagement with devices. This may feel natural to some users. It reduces the need for them to discover it as an explicit feature of the technique. On the flip side, however, users may ascribe shortcut relays to partial reasons only, like input on a companion device. They may therefore be surprised if no cross-device shortcut manifests without a switch in attention. In other words, for inexperienced users, the links between user behavior and system response may initially seem opaque. Training procedures for new users to familiarize themselves with the attention-based transfer mechanism may therefore be critical for adoption.

Related, our study revealed participants' mixed impressions of triggers, which our system supports both explicitly and implicitly. Novice users may benefit from a setting to choose either trigger manner in order to increase the transparency of system behavior.

Another challenge our study uncovered is a lack of communicating system state when content types mismatch. Despite accurate triggering and relaying, a cross-device shortcut may (correctly) not manifest as expected, for example at the UI-element level. Our design rationale was to to display feed-forward where it fits, but users may not be aware of content-type specific input elements or apps (or content-type awareness altogether). Mobile devices diminish such confusion by showing semantic keyboards, which is a concept refined feed-forward previews could build on.

6.2 Learnability

The potential challenges of cross-device shortcuts during first-time use underline the importance of learnability. Based on the experts' insights, the system should transparently render the stages 'trigger', 'relay', 'content-type awareness', and 'feed-forward preview' for new users. This not only makes causal dependencies visible, but also allows users to familiarize themselves with the concept. For example, the system could display feed-forward and feedback for each stage, including highlighting the element on the source device following a trigger (i.e., through a context toast) and animating the outgoing link [3]. Rendering shortcuts depending on availability could also communicate content mismatches, such as grayed-out elements or context-sensitive indicators.

6.3 Robustness & intrusiveness

Our study also emphasized the importance of reliably recognizing (subtle) attention switches, as missed switches (false negative), and accidental activations (false positives) harm fluid operation. One example scenario where accidental activations may occur is when a user wants to evaluate the fit of a photograph (on their phone) in the context of an authored document (on their laptop). For this, the user may switch back and forth between the two devices, which would currently establish shortcuts at every change in glance, each overriding the previous one. False negatives can also occur due to bad lighting or when two devices share the same line of sight.

Equally important is agency over feed-forward display and dismissal. Currently, users can dismiss notifications through 'escape'– however only after the fact, which is an inherent trade-off for feed-forward. To address this, cross-device shortcuts could include a personalization step and settings like 'do not disturb' to configure the display of notifications.

7 LIMITATIONS & FUTURE WORK

While our evaluation shows support for the merits of cross-device shortcuts to enable seamless content transfer, we also learned about characteristics and scenarios that show challenges for their utility. First, we note that our current implementation serves primarily as a proof-of-concept of our interaction technique. As such, our implementation will benefit from further iterations. For instance, more accurate sensing approaches may be integrated enable better gaze tracking and attention inference. Likewise, our implementation assumes a reliable connection between all clients and the server. Intermittent connections would limit practical use.

Beyond implementation-oriented limitations, we believe further exploration of the presentation of feed-forward previews of cross-device interactions may be beneficial. To support user awareness of the information flow within their multi-device workspace, we currently show a preview of the cross-device transfer contents. One interesting alternative may be to present users with a simulation of the transfer result (i.e., the state of the target application when the transfer is executed). Future work may additionally explore enabling real-time modifications of the transfer contents prior to executing the operation.

While our example applications demonstrate the versatility of cross-device shortcuts, we acknowledge that there are contexts where the technique would not provide much utility. For instance, while playing a mobile game, it may not be sensible for the game contents to transfer to a messaging application on a different device. Future work through deployment may help identify when cross-device shortcuts may be beneficial (or not).

Our work currently targets single user multi-device scenarios. In the scenarios we explored, cross-device transfers also only involved one item at a time. In future work, we are interested in investigating the utility of our approach in multi-user scenarios, with a particular focus on how the feed-forward mechanism should be designed. Likewise, we plan to explore feed-forward designs to support a clipboard history or clipboard items from multiple devices. Lastly, given the diversity of devices today, feed-forward designs should be further explored in multi-device ecologies consisting of form factors beyond desktops, phones, and tablets (e.g. smartwatches).

It is additionally important to note that several of our system's features raise difficult privacy challenges. Relying on persistent connection between close-by devices may be problematic in a work environments, where users may still check their personal phones without intending to mix content. One potential solution to this challenge may be to provide end users with access control over shareable contents between devices. Cross-device shortcuts also rely on attention tracking. While this feature is increasingly common in commercial devices (e.g., for authentication), the privacy trade-offs warrant further discussion.

Last, our evaluation was limited to walk-through. This generated interesting insights, but we acknowledge limitations in our approach in validating its usability. In the future, we plan to conduct quantitative comparative studies to further evaluate the usability, efficiency, and technical performance of cross-device shortcuts. Additionally, it would be valuable to include more user perspectives through deployment and longitudinal evaluation.

8 CONCLUSION

We have presented cross-device shortcuts, an interaction technique that leverages a unique combination of attention-driven content transfer, content-type awareness, and multi-level feed-forward to enable deep content links between applications across devices. Cross-device shortcuts explore the integration of contextual and content-type awareness for exchanging content across devices, leveraging an awareness of the user's attention to—upon detecting switches between devices—establish deep shortcut connections from a source app on one device to an app on a target device. Cross-device shortcuts' content-type awareness ensures compatibility between source and target before manifesting. Importantly, cross-device shortcuts visualize the content to be transferred to the target device *before* the user takes action.

We demonstrated cross-device shortcuts in several scenarios, which also formed the basis of an interactive walk-through study with expert researchers and multi-device power users. Our study revealed trade-offs in our approach, notably a conflict between increasing the discoverability of cross-device interaction possibilities and system perceived intrusiveness. We believe that our technique is a step towards facilitating more fluid information transfer between devices, which is a key factor for creating efficient, usable, and productive cross-device ecologies.

REFERENCES

[1] Jessalyn Alvina, Andrea Bunt, Parmit K. Chilana, Sylvain Malacria, and Joanna McGrenere. 2020. *Where is That Feature? Designing for Cross-Device Software Learnability.* Association for Computing Machinery, New York, NY, USA, 1103–1115. https://doi.org/10.1145/3357236.3395506

[2] Olivier Bau and Wendy E. Mackay. 2008. OctoPocus: A Dynamic Guide for Learning Gesture-Based Command Sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (Monterey, CA, USA) *(UIST '08)*. Association for Computing Machinery, New York, NY, USA, 37–46. https://doi.org/10.1145/1449715.1449724

[3] Patrick Baudisch, Edward Cutrell, Mary Czerwinski, Daniel C Robbins, Peter Tandler, Benjamin B Bederson, and Alex Zierlinger. 2003. Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch-and Pen-Operated Systems.. In *Interact*, Vol. 3. 57–64.

[4] Frederik Brudy, Christian Holz, Roman Rädle, Chi-Jui Wu, Steven Houben, Clemens Nylandsted Klokmose, and Nicolai Marquardt. 2019. *Cross-Device Taxonomy: Survey, Opportunities and Challenges of Interactions Spanning Across Multiple Devices.* Association for Computing Machinery, New York, NY, USA, 1–28. https://doi.org/10.1145/3290605.3300792

[5] Frederik Brudy, David Ledo, Michel Pahud, Nathalie Henry Riche, Christian Holz, Anand Waghmare, Hemant Bhaskar Surale, Marcus Peinado, Xiaokuan Zhang, Shannon Joyner, Badrish Chandramouli, Umar Farooq Minhas, Jonathan Goldstein, William Buxton, and Ken Hinckley. 2020. SurfaceFleet: Exploring Distributed Interactions Unbounded from Device, Application, User, and Time. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology.* Association for Computing Machinery, New York, NY, USA, 7–21.

[6] Bill Buxton. 2010. *Sketching user experiences: getting the design right and the right design.* Morgan kaufmann.

[7] Nicholas Chen, François Guimbretière, and Abigail Sellen. 2013. Graduate Student Use of a Multi-Slate Reading System. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 1799–1808. https://doi.org/10.1145/2470654.2466237

[8] Ming Ki Chong, Rene Mayrhofer, and Hans Gellersen. 2014. A Survey of User Interaction for Spontaneous Device Association. *ACM Comput. Surv.* 47, 1, Article 8 (may 2014), 40 pages. https://doi.org/10.1145/2597768

[9] Clippy. 2023. Apple macOS Universal Control: One way to work across your devices. https://clippy.works/. Accessed: 2022-07-22.

[10] Google Cloud. 2020. Detect faces: Cloud Vision API. https://cloud.google.com/vision/docs/detecting-faces?hl=en. Accessed: 2022-03-30.

[11] Sven Coppers, Kris Luyten, Davy Vanacken, David Navarre, Philippe Palanque, and Christine Gris. 2019. Fortunettes: Feedforward about the Future State of GUI Widgets. *Proc. ACM Hum.-Comput. Interact.* 3, EICS, Article 20 (jun 2019), 20 pages. https://doi.org/10.1145/3331162

[12] Tom Djajadiningrat, Kees Overbeeke, and Stephan Wensveen. 2002. But How, Donald, Tell Us How? On the Creation of Meaning in Interaction Design through Feedforward and Inherent Feedback. In *Proceedings of the 4th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques* (London, England) *(DIS '02)*. Association for Computing Machinery, New York, NY, USA, 285–291. https://doi.org/10.1145/778712.778752

[13] Alix Goguey, Sylvain Malacria, and Carl Gutwin. 2018. *Improving Discoverability and Expert Performance in Force-Sensitive Text Selection for Touch Devices with Mode Gauges.* Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3174051

[14] Google. 2021. Google Keep: Free Note Taking App for Personal Use. https://www.google.com/keep/. Accessed: 2022-03-30.

[15] Saul Greenberg and Bill Buxton. 2008. Usability Evaluation Considered Harmful (Some of the Time). In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Florence, Italy) *(CHI '08)*. Association for Computing Machinery, New York, NY, USA, 111–120. https://doi.org/10.1145/1357054.1357074

[16] Peter Hamilton and Daniel J. Wigdor. 2014. Conductor: Enabling and Understanding Cross-Device Interaction. 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, 2773–2782. https://doi.org/10.1145/2556288.2557170

[17] Christian Holz and Frank R. Bentley. 2016. On-Demand Biometrics: Fast Cross-Device Authentication. 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, 3761–3766. https://doi.org/10.1145/2858036.2858139

[18] Kasper Hornbæk and Antti Oulasvirta. 2017. What Is Interaction?. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 5040–5052. https://doi.org/10.1145/3025453.3025765

[19] Edwin L Hutchins, James D Hollan, and Donald A Norman. 1985. Direct manipulation interfaces. *Human–computer interaction* 1, 4 (1985), 311–338.

[20] Apple Inc. 2019. Use Universal Clipboard to copy and paste between your Apple devices. https://support.apple.com/en-us/HT209460. Accessed: 2022-03-30.

[21] Apple Inc. 2021. Apple MacOS Continuity: All your devices. One seamless experience. https://www.apple.com/macos/continuity/. Accessed: 2022-03-30.

[22] Haojian Jin, Christian Holz, and Kasper Hornbæk. 2015. Tracko: Ad-Hoc Mobile 3D Tracking Using Bluetooth Low Energy and Inaudible Signals for Cross-Device Interaction. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 147–156. https://doi.org/10.1145/2807442.2807475

[23] Tero Jokela, Jarno Ojala, and Thomas Olsson. 2015. A Diary Study on Combining Multiple Information Devices in Everyday Activities and Tasks. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3903–3912. https://doi.org/10.1145/2702123.2702211

[24] Gordon Kurtenbach and William Buxton. 1994. User Learning and Performance with Marking Menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, Massachusetts, USA) *(CHI '94)*. Association for Computing Machinery, New York, NY, USA, 258–264. https://doi.org/10.1145/191666.191759

[25] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. 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, 1–17. https://doi.org/10.1145/3173574.3173610

[26] Zhen Li, Michelle Annett, Ken Hinckley, and Daniel Wigdor. 2019. SMAC: A Simplified Model of Attention and Capture in Multi-Device Desk-Centric Environments. *Proc. ACM Hum.-Comput. Interact.* 3, EICS, Article 2 (June 2019), 47 pages. https://doi.org/10.1145/3300961

[27] Tinker Pte Ltd. 2023. Pasteasy. https://download.cnet.com/Pasteasy/3000-2124_4-76191354.html.

[28] Nicolai Marquardt, Till Ballendat, Sebastian Boring, Saul Greenberg, and Ken Hinckley. 2012. Gradual Engagement: Facilitating Information Exchange between Digital Devices as a Function of Proximity. In *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces* (Cambridge, Massachusetts, USA) *(ITS '12)*. Association for Computing Machinery, New York, NY, USA, 31–40. https://doi.org/10.1145/2396636.2396642

[29] Nicolai Marquardt, Frederik Brudy, Can Liu, Ben Bengler, and Christian Holz. 2018. *SurfaceConstellations: A Modular Hardware Platform for Ad-Hoc Reconfigurable Cross-Device Workspaces.* Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3173574.3173928

[30] Nicolai Marquardt, Nathalie Henry Riche, Christian Holz, Hugo Romat, Michel Pahud, Frederik Brudy, David Ledo, Chunjong Park, Molly Jane Nicholas, Teddy Seyed, Eyal Ofek, Bongshin Lee, William A.S. Buxton, and Ken Hinckley. 2021. AirConstellations: In-Air Device Formations for Cross-Device Interaction via Multiple Spatially-Aware Armatures. 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, 1252–1268. https://doi.org/10.1145/3472749.3474820

[31] Microsoft. 2018. Use Clipboard in Windows. https://support.microsoft.com/en-us/windows/clipboard-in-windows-c436501e-985d-1c8d-97ea-fe46ddf338c6. Accessed: 2022-03-30.

[32] Don Norman and Bruce Tognazzini. 2015. How Apple is giving design a bad name. *Co. Design* (2015).

[33] Donald A Norman. 1986. Cognitive engineering. *User centered system design* 31 (1986), 61.

[34] Donald A Norman. 1991. Cognitive artifacts. *Designing interaction: Psychology at the human-computer interface* 1, 1 (1991), 17–38.

[35] Roman Rädle, Hans-Christian Jetter, Jonathan Fischer, Inti Gabriel, Clemens N. Klokmose, Harald Reiterer, and Christian Holz. 2018. PolarTrack: Optical Outside-In Device Tracking That Exploits Display Polarization. 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, 1–9. https://doi.org/10.1145/3173574.3174071

[36] Jun Rekimoto. 1997. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology* (Banff, Alberta, Canada) *(UIST '97)*. Association for Computing Machinery, New York, NY, USA, 31–39. https://doi.org/10.1145/263407.263505

[37] Dan Saffer. 2010. *Designing for interaction: creating innovative applications and devices.* New Riders.

[38] SAMSUNG. 2021. Samsung Flow. https://www.samsung.com/us/support/owners/app/samsung-flow. Accessed: 2022-03-30.

[39] Stephanie Santosa and Daniel Wigdor. 2013. A Field Study of Multi-Device Workflows in Distributed Workspaces. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Zurich, Switzerland) *(UbiComp '13)*. Association for Computing Machinery, New York, NY, USA, 63–72. https://doi.org/10.1145/2493432.2493476

[40] Marcos Serrano, Barrett Ens, Xing-Dong Yang, and Pourang Irani. 2015. Gluey: Developing a Head-Worn Display Interface to Unify the Interaction Experience in Distributed Display Environments. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark) *(MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 161–171. https://doi.org/10.1145/2785830.2785838

[41] Jeffrey Stylos, Brad A. Myers, and Andrew Faulring. 2004. Citrine: Providing Intelligent Copy-and-Paste. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology* (Santa Fe, NM, USA) *(UIST '04)*. Association for Computing Machinery, New York, NY, USA, 185–188. https://doi.org/10.1145/1029632.1029665

[42] Peter Tandler, Thorsten Prante, Christian Müller-Tomfelde, Norbert Streitz, and Ralf Steinmetz. 2001. Connectables: Dynamic Coupling of Displays for the Flexible Creation of Shared Workspaces. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology* (Orlando, Florida) *(UIST '01)*. Association for Computing Machinery, New York, NY, USA, 11–20. https://doi.org/10.1145/502348.502351

[43] Jo Vermeulen, Kris Luyten, Elise van den Hoven, and Karin Coninx. 2013. *Crossing the Bridge over Norman's Gulf of Execution: Revealing Feedforward's True Identity*. Association for Computing Machinery, New York, NY, USA, 1931–1940. https://doi.org/10.1145/2470654.2466255

[44] Simon Voelker, Sebastian Hueber, Christian Holz, Christian Remy, and Nicolai Marquardt. 2020. GazeConduits: Calibration-Free Cross-Device Collaboration through Gaze and Touch. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3313831.3376578

[45] Paweł Woundefinedniak, Lars Lischke, Benjamin Schmidt, Shengdong Zhao, and Morten Fjeld. 2014. Thaddeus: A Dual Device Interaction Space for Exploring Information Visualisation. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational* (Helsinki, Finland) *(NordiCHI '14)*. Association for Computing Machinery, New York, NY, USA, 41–50. https://doi.org/10.1145/2639189.2639237

[46] Ye Yuan, Nathalie Riche, Nicolai Marquardt, Molly Jane Nicholas, Teddy Seyed, Hugo Romat, Bongshin Lee, Michel Pahud, Jonathan Goldstein, Rojin Vishkaie, Christian Holz, and Ken Hinckley. 2022. Understanding Multi-Device Usage Patterns: Physical Device Configurations and Fragmented Workflows. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 64, 22 pages. https://doi.org/10.1145/3491102.3517702