# Escapement: A Tool for Interactive Prototyping with Video via Sensor-Mediated Abstraction of Time


Molly Jane Nicholas*
University of California, Berkeley

Nicolai Marquardt†
Microsoft Research, UCL

Michel Pahud
Microsoft Research

Nathalie Henry Riche
Microsoft Research

Hugo Romat
Microsoft Research

Christopher Collins
Microsoft Research

David Ledo
Microsoft Research

Rohan Kadekodi
Microsoft Research

Badrish Chandramouli
Microsoft Research

Ken Hinckley
Microsoft Research


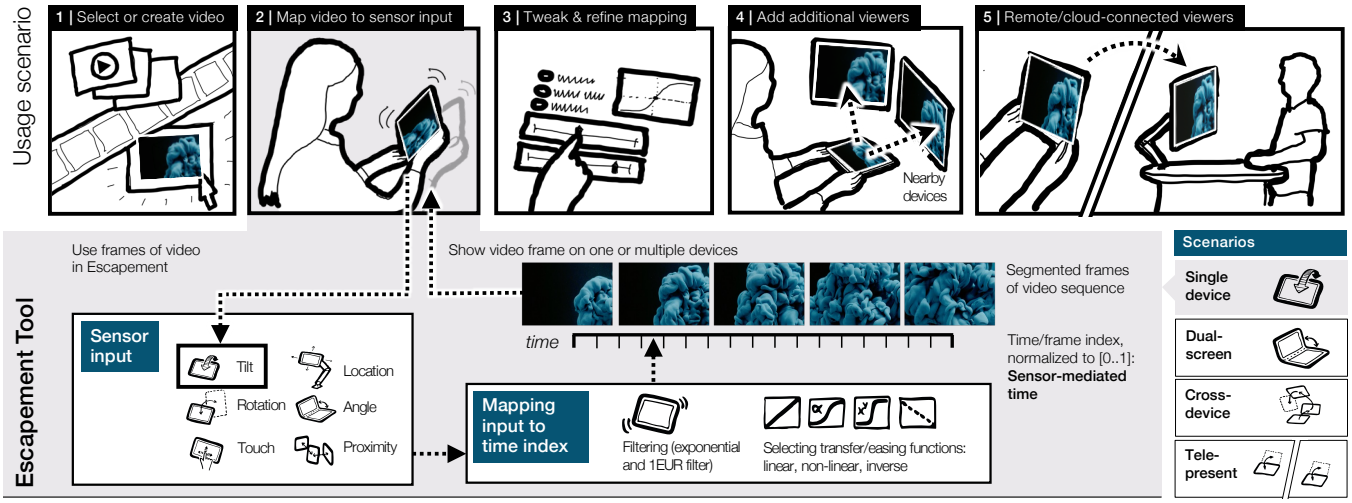


**Figure 1: Escapement is a prototyping tool that reifies video snippets as sensor-mediated interactive prototypes, for screen-based applications, and across one or more devices. The tool flexibly maps a variety of real-time sensor inputs (such as tilt, motion, or touch) to the time index of a pre-recorded video (or series of still images). This empowers designers to work directly with "time as a design material" in the prototyping process, exploring the feel of an interaction in response to sensor data and corresponding visual feedback.**


## ABSTRACT

We present Escapement, a video prototyping tool that introduces a powerful new concept for prototyping screen-based interfaces by flexibly mapping sensor values to dynamic playback control of videos. This recasts the time dimension of video mock-ups as sensor-mediated interaction.

This abstraction of time as interaction, which we dub *video-escapement prototyping*, empowers designers to rapidly explore and viscerally experience direct touch or sensor-mediated interactions across one or more device displays. Our system affords cross-device and bidirectional remote (tele-present) experiences via cloud-based state sharing across multiple devices. This makes Escapement especially potent for exploring multi-device, dual-screen, or remote-work interactions for screen-based applications.

We introduce the core concept of sensor-mediated abstraction of time for quickly generating video-based interactive prototypes of screen-based applications, share the results of observations of long-term usage of video-escapement techniques with experienced interaction designers, and articulate design choices for supporting a reflective, iterative, and open-ended creative design process.



*molecule@berkeley.edu
†nicmarquardt@microsoft.com





*CHI '23, April 23–28, 2023, Hamburg, Germany*



ACM ISBN 978-1-4503-9421-5/23/04.
https://doi.org/10.1145/3544548.3581115

## KEYWORDS

video prototyping, creativity support tools, cross-device interaction, sensor-mediated interaction

**ACM Reference Format:**

Molly Jane Nicholas, Nicolai Marquardt, Michel Pahud, Nathalie Henry Riche, Hugo Romat, Christopher Collins, David Ledo, Rohan Kadekodi, Badrish Chandramouli, and Ken Hinckley. 2023. Escapement: A Tool for Interactive Prototyping with Video via Sensor-Mediated Abstraction of Time. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3544548.3581115

## 1 INTRODUCTION

Designers leverage a range of tools when building prototypes of interactive systems, from paper and foam core, to video prototypes, to fully-functional digital prototypes. But a major disadvantage of video-prototypes is that they are not interactive – they are limited to visual playback, where a designer (or usability participant) can only "watch" rather than viscerally feel and enact the experience. Designers additionally value tools that require minimal code generation [36], especially early on in the design process. Tools used and artifacts created through this exploration stage shape the final designs and help designers define the "scope" of exploration [39] in an evolving, reflective "conversation with materials" [11, 12, 53]. Early ideation tools should empower designers to explore multiple ideas, try alternatives, and iterate until the design *feels* right.

As part of our goal to empower designers with new capabilities and fluid design exploration, we introduce *Escapement*, a tool that leverages recorded video as an interactive design material, allowing designers to "rapidly generate multiple alternatives [and] explore their implications" [54]. This tool enables *video-escapement prototyping*, where designers take a video snippet, map it to sensor data, and replay the recording in response to real-time user input. On-device (or external) sensors then map dynamic input information – including touch, tilt, and motion paths – to on-screen changes.

Our tool can stream sensor data between a plurality of devices, allowing designers versatility and flexibility in terms of input modality, input degrees-of-freedom (DoF's), transfer functions, and the graphical mirroring or time-reversal of on-screen effects in each viewport. Escapement provides designers with the ability to *engage with time as a first class design material* through easing functions (slow-in/slow-out), reversing the playback time-sequence of the video, auto-playing video sequences in a loop, or touching the screen to stop (freeze / clutch) or start (initiate or resume) playback in response to sensor data. By providing interactive, sensor-mediated control of the temporal dimension of videos, Escapement offers designers direct, reversible, and manipulable access to this otherwise intangible design element of *time*.

Designers can thus engage in *embodied exploration* [12, 30] by modifying the input modality and DoF's, motion paths, and transfer functions for screen-based applications – that is, the kinesthetic "feel" of sensed inputs in response to corresponding visual feedback on one or more displays. And because Escapement supports multiple devices, input from one device can trigger changes on any of the other network-connected displays. Further, in addition to the temporal dimension (which can be played either forward or backward in time), the spatial dimension can readily be manipulated on each screen, such as by reflecting the video left-right or flipping it upside-down. Quick control over such symmetries lets designers experiment with short video clips (typically of 3-30s duration) and mold them into the building blocks of interaction techniques with appropriate feedback, where graphics flow across or react to multiple devices, screens, and viewports in a synchronized and spatially corresponding manner. Designers are thus empowered to experiment with live, interactive, cross-device experiences, to explore stimulus-response compatibility of input and feedback, and to rapidly develop richly interactive mixed-fidelity prototypes [45].

Escapement represents a step towards richer video prototyping methods. Researchers have explored myriad ways to expand the design space of video prototyping tools, for example by designing tools that support the reuse of filmed content [35] or that enable designers to work directly with on-device sensors [33]. Building on a similar philosophy, our system elevates pre-recorded video snippets – whether from keyframes, mock-ups, captured video, screen recording, stock footage, a series of still frames (PNG images), or output from design and presentation tools (such as Figma or Powerpoint) – into an interactive prototype that supports *interaction-driven animations* [34]. This type of interactivity has been referred to as *user-in-the-loop behaviors* [19, 33], where system output is driven directly by continuous user input (see Figure 2).

In sum, our work contributes the following:

- Establishment of the general notion of *video-escapement prototyping*, which operationalizes time as a design material by recasting the temporal dimension of video clips as designer-controlled (or test-user-controlled) interaction.
- The design and implementation of the Escapement tool, which realizes *video-escapement* prototyping as a means to prototype with sensor-mediated abstractions of time on one or more screen-based devices without the need to code or understand complex software -development concepts.
- Observation and reflection on long-term use of Escapement, with insights including the ways video content shapes the interaction designs created, the design implications of being able to rapidly swap sensors and degrees-of-freedom, and using time as a design material.

The usage strategies noted above evolved and emerged over 3+ years of continuous usage across more than 20 interaction designers, research interns, visiting scholars, and other collaborators. Our hope is that by articulating this approach, and describing some of its key characteristics, we can empower other designers to adopt it into their design processes where appropriate, as well as to potentially inspire a new generation of design tools that can co-opt and build upon Escapement's core conceit: the abstraction of time from video, such that time becomes a design material that can be explored and enacted through direct, embodied experimentation.

In the following sections, we first review Related Work, then describe the rationale behind the Escapement Tool itself, including key Design Decisions as well as the Technical Architecture of our implemented system. We then walk through an Escapement Usage Scenario, followed by a characterization of some Design Strategies supported by the video-escapement prototyping design method.

We then reflect on the Escapement Design Iteration and Evolution under longitudinal use, with a Discussion of some strengths and limitations of the tool in the broader context, leading finally to a Conclusion and Future Work.

## 2 RELATED WORK

Escapement is a prototyping tool that supports rapid, iterative, and embodied interaction design of screen-based interfaces on one or more devices in a way that builds on *video prototyping* and *smoke and mirrors* [5]. As a result, prototypes created with Escapement can take on different roles in the interaction design process. Because video frames are remapped to sensor data, designers have to work with time in mind, which differentiates it from past systems using *smoke and mirrors* and *screen poking*. Lastly, Escapement can work with one or more devices, and thus builds on past work in multi-screen and cross-device interaction.

### 2.1 Background: Prototyping in Interaction Design

Sketching and prototyping are both recognized as key activities in the interaction design literature [5, 9, 10, 22, 26, 34, 39, 45, 55], often distinguished based on the goal at hand (i.e., expanding ideas versus defining future implementations) [5]. While prototypes can be discussed as early externalizations of a design, they are better described as explorations that answer a specific question in the process of better understanding the problem at hand: they are filters and manifestations of design ideas [39]. Under this perspective, sketching and prototyping seem indistinguishable, more geared towards the role they play in the stage of the design process (fidelity), and their degree of sophistication (resolution) [22]. Thus, we refer to these collective activities as prototyping.

Prototyping is key in design practice to achieve problem construction [26], a process in which problem definition and the solution co-evolve [10] through systematic explorations that answer specific questions [39, 55]. There are many frameworks to explore these types of questions, including implementation, look-and-feel, and role [22]; content, form, and behaviour [9]; and structure, behaviour, and usage [34].

There are many techniques and tools that can be employed to explore a design space. A common approach is *wireframing*, where individual panels illustrate a user interaction playing out over time [16]. This assumes a set of well-defined states, which transition between explicit actions, and are best suited to illustrate the flow of an application [9]. The Wizard Of Oz technique [28], on the other hand, relies on a behind-the-scenes expert controlling the interaction to explore the fidelity of a particular experience [5]. Video prototyping [41, 42] can capture both types of non-interactive software demonstrations, providing added insight of how an interaction might appear.

With these approaches, it is possible to experience some of the elements of the interaction, but they are not suited for exploring interactive behaviour —*the animacy of an effect driven by the interaction as it happens in real time*, and are tightly coupled with appropriate feedback. This is defined in the literature as *user-in-the-loop behaviours* [19] or *interaction-driven animation* [34]. For these types of behaviours, there is a need for tools and approaches that can support *thinking and doing* [12] and enable *reflection-in-action* [53] through embodied externalized experimentation [21, 24, 30].

### 2.2 Designing with Time in Mind: Video Prototyping

The advent of computer animation enabled the creation and consumption of time-based media. Perhaps the earliest example of playing back an animation coupled to the motion-dynamics of an input source, via pen-driven timing of an animation, appeared in Baecker's Genesys system [1]. Over time, physical approaches to animation migrated into computer graphics, bringing forth the notion of *keyframes* as core visuals supported by other transitional visuals over time [4]. Animators ported traditional methods such as frame-by-frame and *tweening* – interpolating between keyframes with parameters that change as a function of time – into digital animation. These authoring paradigms remain core in supporting the design of new user experiences through video editing software (e.g., Da Vinci Resolve [1], Adobe Premiere [2]) as well as animation software (e.g., Adobe Animate [3] (formerly Flash), and ToonBoom Harmony [4]).

One way of expressing time-based interactions is through the practice of video prototyping, which consists of capturing and showcasing interactions through captured videos or animations [41, 42]. Building prototypes with *rich interactivity* and *visual refinement* – even while minimizing the focus on *breadth and depth of functionality* [45] – still typically comes at a high engineering cost [45], especially compared with more rapid tools such as paper and foam core. While video is a powerful and widely-used tool [49], a major disadvantage of video prototypes is that they are not interactive - they are merely a visual experience that does not support *user-in-the-loop* animations, where system output is mapped directly to continuous user input [19]. Video prototyping can provide the best aspects of 1) rapid prototyping of 2) highly expressive design concepts [35, 41, 42, 57].

Video prototyping can be understood as an expressive and flexible form of sketching ideation [5] which enables designers to explore a particular design space [39] either as part of an individual creative exploration, or a collaborative evaluation and iteration process [42]. However, video prototypes are typically used to capture a Wizard of Oz-style performance by a designer, relying on video-editing trickery to convey the design [35, 41, 42]. This experience is largely visual and limited to linear-time playback of the resulting content. We extend video prototyping by creating a design tool that supports interacting with the prototype, allowing the video to dynamically respond to actions in an embodied way. Through Escapement, a designer can rapidly prototype an interaction on the target device – mapped in real time to the actual sensor DoF's of interest – to quickly try multiple alternatives, iterate, and get the right feel.

*Compositing*, whether digital or physical, is another editing technique borrowed from filmmaking that combines visual elements

[1] https://www.blackmagicdesign.com/products/davinciresolve/
[2] https://www.adobe.com/products/premiere.html
[3] https://www.adobe.com/products/animate.html
[4] https://www.toonboom.com/products/harmony

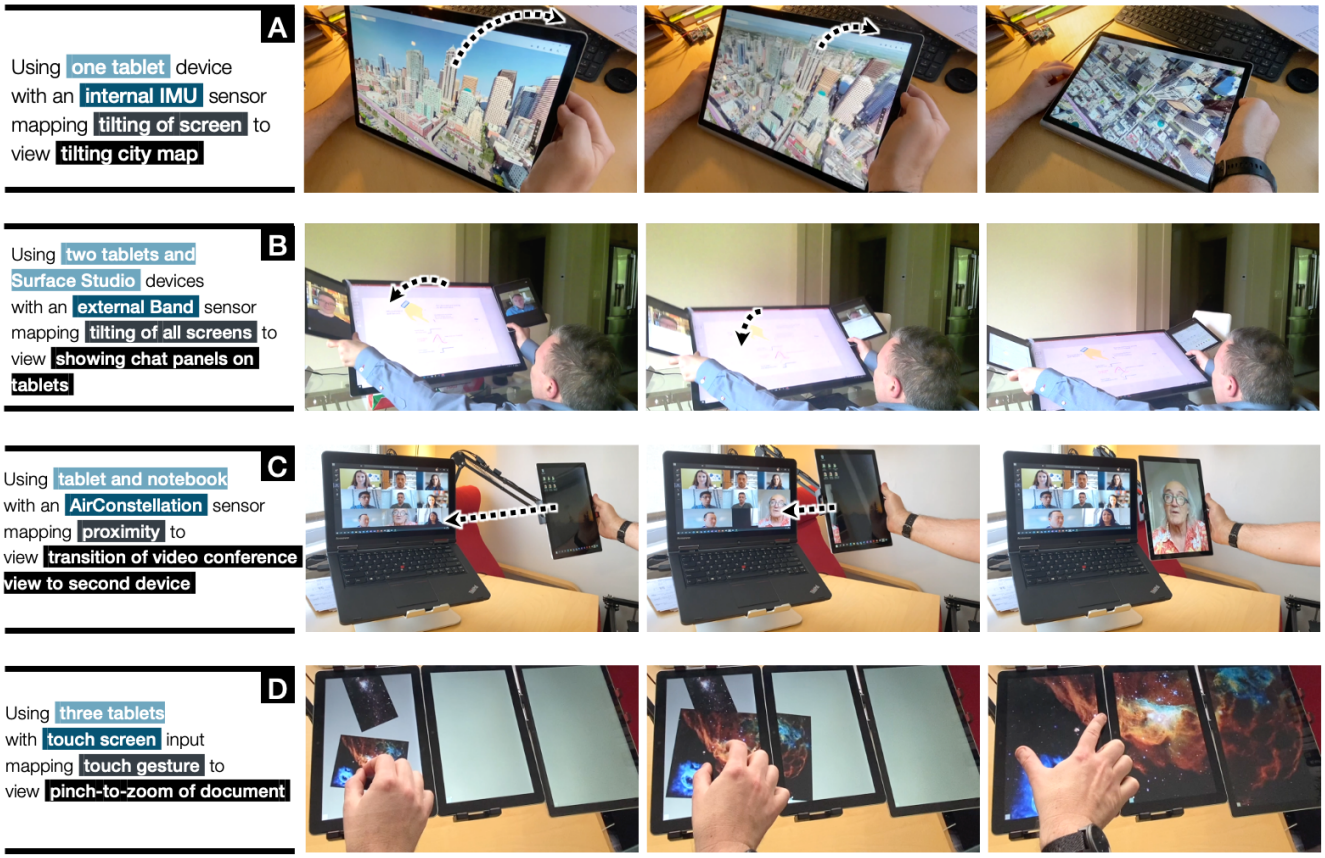


**Figure 2: Four examples of designs generated with the Escapement tool. A) A tablet device running both the Sensor Mapper and the Viewer. A tilting motion changes the spatial perspective view of a city. B) A designer attaches an external IMU sensor to a Surface Studio to detect tilt motion. Viewers are open on the Surface Studio, and two additional tablets, which all respond to the tilting motion. The tablets both transition from video content to chat panels as the Surface Studio is tilted down. C) An AirConstellations-enabled tablet [43] allows a designer to use sensed proximity to control the behaviour of a video-conferencing application. D) The Sensor Mapper running on a single tablet (on the left) and an animation split into portions that run on the Viewers on the other two tablets creates the illusion of cross-device zooming via a pinch gesture. Note that all interfaces shown are captured pictures that were ported into the tool - the tool does not interact directly with other applications.**

from multiple sources into a single final image or video [5]. This ability to draw on multiple independent sources allows for a *bricolage* approach [40, 56], that is, creation from a diverse range of design materials that happen to be available. Within the context of interaction design, bricolage has been described as related to but distinct from Schön's reflective conversation with materials [53], and additionally centers around constructing an interactive prototype rather than a sketch [56]. Escapement embraces this notion of co-opting video materials at hand (whether screen recordings of existing applications, stock video clips that capture the gestalt of an idea, "found" video encountered on the web, or animations/transitions from presentations that can be exported as short video clips) and bringing them into our tool to experience interactively within minutes – a bricolage of video.

[5]https://en.wikipedia.org/wiki/Compositing

## 2.3 Interactive Prototypes: Smoke and Mirrors and Screen Poking

To achieve *interaction-driven* animations [19, 34], it is necessary for a system to dynamically interpret and drive the responses to people's actions. Hartmann [19] refers to two types of animation playback as a function of input: *one-shot animations*, which play after an action is performed, such as pressing a button, and *user-in-the-loop behaviors*, where "continuous input drives the behaviour" [20]. Prototyping tools can leverage both types of animations, as demonstrated by Astral [33] and Monet [38].

One way to create interaction-driven animation experiences is through what Hartmann [19] refers to as *"screen poking"*, in which sensors are mapped to a computer's mouse and keyboards (e.g., BOXES [23] to make the sensors instantly interactive). While typical examples of screen-poking require interacting directly with a desktop computer, Astral [33] closed the screen-poking loop by

allowing designers to map mobile sensor data to mouse and keyboard inputs on a computer, and having the computer stream a portion of the screen to the mobile device. Alternatively, systems can provide direct authoring to map inputs to animations, as done by Monet [38], Kitty [27], and Enact [37].

Authoring environments that provide "user-in-the-loop" control can be difficult to create. One common authoring approach is through *"smoke and mirrors"* prototyping [5], where trying a prototype can feel fully functional while complex workarounds to make it happen are hidden, as done by Chameleon [14]. In many cases, the easiest way to do this is to leverage existing infrastructures [47] and simply work to repurpose them. Olsen [47] made this case in the context of toolkits, where for instance, a stylus could work with the existing mouse cursor infrastructure and achieve a basic level of functionality to interact with an operating system without requiring extensive implementation efforts. With smoke and mirror prototypes, it is possible to create rich experiences that appear functional at a fraction of the effort. With Escapement, an extensive circle of collaborators has been able to create conceptual prototypes that explore broad design spaces [43, 51] while also requiring minimal additional engineering investment.

While Astral's approach could work with videos in an editor, the video itself was not the design material [33]. One can map a video's frames to a particular gesture, as shown by DIMP [13]. The advantage of working directly with video is that video-creation is decoupled from interaction; a designer can use any "found" pre-existing video, or generate a new one, as necessary to prototype a particular experience. Thus, to use sensor-to-video mappings for prototyping, these mappings must be changeable on the fly for quick testing and experimentation. This principle shaped our design of Escapement and is how a designer can quickly turn rapidly-made videos into multiple interactive experiences, a smoke and mirrors with a single path of interaction where the output is only bound to the expressiveness on the video itself. Like Astral [33] and Microsoft Blend [6], the inputs can be completely modified by using animation techniques such as easing functions or reversals [50] therefore changing the quality of how an interaction feels as it takes place.

## 2.4 Designing for Multi-Screen and Cross-Device Experiences

Given Escapement's strength in working with multiple viewports distributed across one or more devices, we situate our work within multi-screen and cross-device interaction design. As multi-device usage continues to become the norm for both work and play, there is increasing need for designers and engineers to generate novel interaction techniques [18, 25, 31, 58], application scenarios [2, 17] and mental models that will best support cross-device experiences. Researchers have designed systems for distributing application interfaces across multiple devices [58], but without lengthy investment in recreating such a setup, other designers may find it difficult to iterate on the design concepts.

Likewise, novel and emerging form-factors (such as multi-screen or foldable devices) often incorporate new DoF's of sensing as well as interaction techniques coordinated across displays. For example, the designers of Flecto [29] recognized the unique challenges of designing for a foldable interface, and created a prototyping system for quickly iterating on designs with minimal engineering needs, or even a hardware prototype. By contrast, Escapement offers a generalizable approach to rapid-prototyping applicable to foldable/dual-screen devices as well as a wide variety of other continuous, sensor-mediated interactions across displays situated on one or more devices.

Recent research on cross-device interaction and multi-device systems identifies the particular challenges of designing for multiple devices at once [3], such as multiple displays responding simultaneously, synchronizing video playback across different video displays, or sharing sensor data input across distributed devices. Tools that support rapid interactive prototyping across multiple device ecosystems remains difficult and time-consuming – yet represent a major strength of Escapement , which can share state between multiple cloud-connected instances of the tool.

# 3 THE ESCAPEMENT TOOL

Escapement transforms video prototypes into interactive experiences by normalizing both sensor input and video timelines to logical (unitless) dimensions between 0 and 1, to allow for a high degree of flexibility, "plug-and-play" exploration, and sensor-agnostic designs. As designers simultaneously define and explore a new design space [39], they need to engage in contextualized *design by enaction* [37] to "support the design and implementation of an interaction through enactments of the interaction as a rapid, active and contextualized medium for design." Escapement supports this through four primary design decisions: 1) sensor-and-output agnosticism; 2) normalizing video timelines; and 3) supporting multi-device formations through 4) streamed sensor data.

## 3.1 Design Decisions

*3.1.1 Sensor-and-Output Agnosticism.* The use of sensor data in interaction design can be understood along a spectrum from highly input/output agnostic (lower expressive match, high flexibility, and plug-and-play) to sensor-specific (high expressive match / less flexibility). In contrast with Astral [33] – an example of an extremely sensor-specific design – our sensor-agnostic design enables rapid transformation of both input (sensor data) and output (one or more video sequences). In the Escapement tool, simple menus, parameter sliders, and checkboxes let the designer quickly try out various sensors or DoF's as inputs; one can even specify a desired range of motion by demonstration, for example. This agnosticism to the source of the input sensor lets designers reflect-in-action by actually trying numerous sensors and mappings, as well as to gain direct experience with the strengths and weaknesses of perhaps-unfamiliar sensor modalities. But this agnosticism is also reflected in the output, for example by using the same sensor stream to drive multiple viewports; but even those viewports can further transform the output, such as by mirroring it left-right or up-down.

*3.1.2 Normalized Input and Output Dimension.* Escapement allows designers to transform short MP4 video snippets into an interactive animation by slicing the video into a sequence of still frames. These frames can then be replayed in response to streamed sensor data. The tool normalizes the length of a video along an abstract 0 (start) to 1 (end) temporal dimension, and allows designers to replay any

[6] https://en.wikipedia.org/wiki/Microsoft_Blend

sequence of frames along an independent input (spatial) dimension, also from $0 \rightarrow 1$. This simple design decision yields a key strength of Escapement: by ensuring that all videos can be normalized to the same logical $0 \rightarrow 1$ mapping, we provide designers with a *path of least resistance* [46] to sensor-agnostic, user-in-the-loop [19, 33] animation design. This feature also makes it easier to reverse either the spatial input or the temporal output (i.e. with a $-1 \rightarrow 0$ mapping).

*3.1.3 Multi-device Support.* Animations can be replayed on either the same device as the sensor input, or across multiple viewports. Additional viewports can appear on the same device, on another display in multi-screen setups, or on another networked (co-located or remote) device via UDP multicast or Azure cloud streaming. Different video sequences can play on different viewports, in response to the same sensor data (in our current implementation, video content is not streamed, only sensor data; each Escapement instance imports the video still-frames from local storage).

This notion of local or remote viewports allows designers to design experiences that span multiple devices, meaning that they are synchronized, flow across, or otherwise react to and show visually connecting feedback between one another. It is also possible to simultaneously run multiple instances of Escapement with separate viewports that respond to additional sensor data or touch input sources: the screen of any Viewer can serve as an input to drive other animations.

*3.1.4 Streamed Sensor Data.* Escapement captures and streams data from built-in or externally attached sensors. Escapement then replays sequenced video frames in response to the chosen degree-of-freedom of sensor data. This allows designers to use sensor input to control animations (including sensed position, movement along a path, orientation changes, left-right or up-down touchscreen input, etc.). Likewise, touch inputs, relative (mouse) cursor motion, or even a simple slider can provide virtual "sensor" input.

Together, the above functionality enables designers to leverage video as a first-class design material *for interaction* during early-stage prototyping.

## 3.2 Escapement Technical Architecture

Escapement enables designers to quickly explore novel interaction techniques with real sensor data. Escapement consists of two primary applications: the *Sensor Mapper*, and the *Viewer*. We also briefly discuss the *Slicer*, a separate utility that pre-processes videos into an ordered sequence of still frames consumed by the Escapement Viewer.

*3.2.1 Sensor Mapper.* The sensor mapper provides a central hub for managing streaming sensor data across multiple devices. The sensor mapper manages the sensor data, applies a variety of possible designer-selected filters, including the one-euro filter [6], and redirects it as part of a UDP packet stream. Escapement can use UDP multicast to stream sensor data across connected devices on a local network, or alternatively, we also implement communication through a custom shared memory abstraction using a server running on a virtual machine (VM) hosted on Azure. The cloud version allows us to work beyond local networks, prototype experiences that involve remote users, overcome firewalls and router settings preventing UDP, or indeed to prototype and experience interactive demonstrations that span any two cloud-connected devices. As shown in Figure 3, detailed settings allow designers to specify how the live sensor data is processed before being mapped to the animations (e.g., filter thresholds, cutoff values), or during playback (e.g., time-reversed playback, spatial flipping or mirroring of the animation still-frames, or selecting a subset of the animation sequence).

*3.2.2 Viewer.* The prototyping Viewer client software displays the different animated application sequences on each device. Each viewer client receives the real-time sensing metadata (as part of a local loopback, local-network UDP multicast, or cloud packet stream) from the Escapement platform, transforming normalized input into the abstract time dimension. This data takes the form of a video frame index based on sensing data (such as tilt angle), which the Viewer uses to display the correct frame of the video sequence. This allows the Viewer to dynamically animate the application interface in response to changing inputs such as touch, or device motion along a particular path or degree of freedom. The Viewer also directly supports simple options for reverse-time playback, mirroring the video spatially (i.e. left-right or up-down), or defining the viewport scaling function (Fill, Uniform, Uniform-to-Fill, or None). These options make it easy to experiment and play with the response on any particular display (viewport) individually to correspond to the user's input and feedback on other displays (please refer to the video figure of the auxiliary material for demonstrations of how surprisingly quick and effective this can be).

| Network | Mean | Median | StdDev | Min | Max |
|---|---|---|---|---|---|
| Corporate Wi-Fi | 10.8 | 9.8 | 3.6 | 8.3 | 28.6 |
| Home Wi-Fi | 14.9 | 13.1 | 7.1 | 9.6 | 59.9 |
| Mobile LTE | 56.1 | 48.1 | 28.1 | 34.3 | 190.2 |

**Table 1: Results of network latency test when using cloud synchronization of states in Escapement (all measurements in milliseconds (ms); in each network condition, 100 messages were sent over 10 minutes through Azure Cloud).**

For each Sensor Mapper, a designer may connect one or more Viewers running on either the same machine (to minimize latency), or across multiple networked machines. The latter may introduce some latency; yet while latency can impact user experience [44], network delay is an inherent quality of cross-device interaction, and therefore should be apparent in early-stage prototypes to reflect the reality of time-lagged feedback. Most recently, motivated primarily by the hybrid and remote-work scenarios of the pandemic, we have added the ability to share sensor data and other animation states via the cloud (implemented on top of the FASTER key-value store [7] and the CRA library for virtual connectivity across clients [52]), which allows us to connect prototypes across remotely-connected devices in a performant manner. We measured the latency of synchronizing states through the Azure Cloud in different network conditions (summarized in Table 1). We tested the network round-trip time (divided by two) of 100 messages in each network condition, sent over 10 minutes by the Sensor Mapper through the cloud services to the Viewer and relayed back.

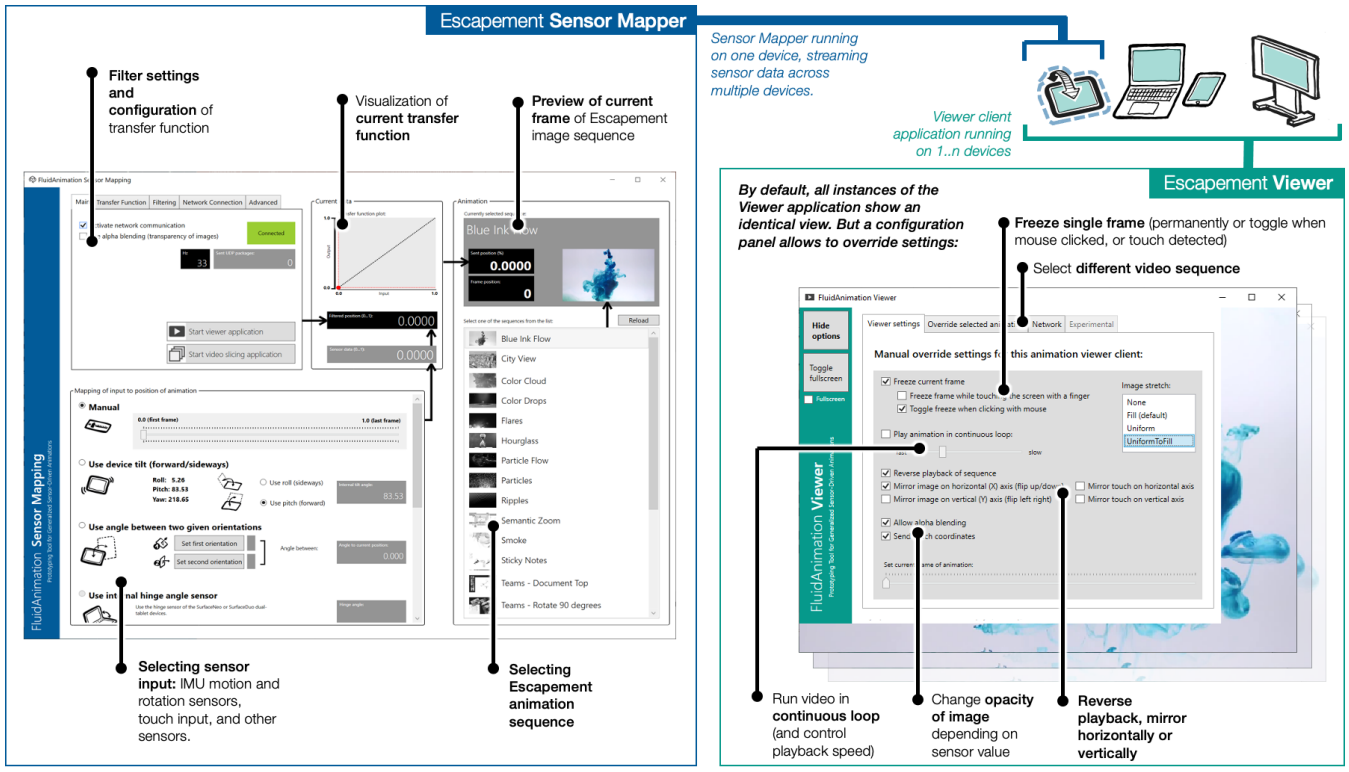


**Figure 3: The UI for both the Sensor Mapper (Left) and the Viewer (Right). The Sensor Mapper interface allows the designer to select what sensor data is read (IMU rotation and motion, touch, etc.), and how that sensor data is updated (transfer functions, cutoffs). Streamed sensor data is visualized here, before any Viewer is opened. It's also possible to directly launch a Viewer for a particular animation sequence from the panel on the right. The Viewer provides access to animation controls such as time-reversing, image mirroring, continuous play, freeze (permanently or on interaction), and image opacity. Designers can thus quickly update the animation replaying in response to the same sensor.**

These measurements give an estimate of network latency, but do not include latency caused by sensing hardware, filtering, or visual rendering of the animation. The average latency ranges from *10.8 ms* (in the corporate Wi-Fi network) to *56.1 ms* (mobile LTE connection). Depending on connection quality, maximum latency measured in our tests reached *190.2 ms* (with the LTE connection). This means that – depending on the quality of the network connection – in some cases network latency might lead to noticeable lag in how animations shown in the viewer respond to sensor input, and this needs to be taken into account when creating animation sequences where fast response time is critical.

*3.2.3 Slicer.* All videos used in Escapement are pre-processed with the Slicer tool, which extracts a sequence of still-image frames from the video. This image sequence is then used by the Viewer to show the corresponding frame for the current time index. There is no set limit on the length of videos (beyond local memory capacity); the Escapement front-end imports videos of unspecified length, and the designer can decide how many discrete still-frames the video is distilled to. (The default for short videos is 60 frames; for longer videos – suitable for fine-grained sensor dimensions such as accurate touchscreen coordinates across an entire screen – the designer can set the Slicer to extract 20 frames per second, for example). A designer can also directly load image sequences from other applications (skipping the video-import step), such as sequences of renderings out of 3D animation tools, or frame-by-frame animations exported from slideshow presentations.

Escapement is not picky about the number of still-frames. Designers can directly edit, delete, duplicate, or add to these preprocessed still-frames if desired. For example, to emphasize a keyframe in a sensor-mediated transition that should be more salient, that particular image can be duplicated multiple times and inserted into the sequence. The designer can also easily delete any undesired images from the image folder to tweak the sequence and its interactive-playback timing. Escapement simply uses the sort-order based on file names (typically a numbered sequence) to determine the order of frames, making it easy to tweak individual frames in this manner if desired without generating a new video and re-slicing it.

*3.2.4 Summary.* By segmenting video into sequences of still frames, and providing detailed settings for specifying how the live sensor tracking data is mapped to the animations, our tool allows users to create interaction-driven animations. To do this, the designer (1) selects the video or other source material and converts this into a

sequence of still images, (2) selects the sensor input to control the animation (such as sensed position, movement along a path, orientation changes, etc.), (3) optionally tweaks the transfer function (e.g., linear, non-linear, ease-in/out) that controls the mapping of sensor input to animation frame, and (4) specifies which video sequence to show on which viewport (device or screen). We illustrate all of these steps in the following usage scenario.

## 4 ESCAPEMENT USAGE SCENARIO

While there are many different strategies that designers might use when working with Escapement, we first describe a basic walk-through for generating an interaction design with Escapement. Let's follow the design process of Chichima, a designer working on a new project that includes sharing 3D model content into video-conferences during remote work. This scenario is a distilled version of how we've observed designers using the tool throughout the years of its existence.

***Creating Video Clip.*** Since Chichima is working with 3D models, she naturally gravitates to experimenting with spatial interaction using her tablet's in-built motion sensors. She quickly records a screen-capture as she uses her mouse to rotate a 3D avatar model already at hand. Chichima then splices out just a small clip of the 3D avatar model pivoting between two views. Then, using the Slicer tool she imports the video into Escapement, creating a sequence of "flipbook" images that will play back in response to her sensor input.

***Map Video to Sensor Input.*** Chichima can now start playing with and gain direct hands-on experience with how her short video of the 3D avatar rotation feels with different inputs and sensor mappings. She opens both the Sensor Mapper and the Viewer on a single device - her tablet. She selects the "tilt sensor" option, and within seconds she's able to see the 3D model perspective shift in response to her physically adjusting the left-right tilt angle of the tablet.

***Tweak and Refine Sensor Mapping.*** By trying out her proposed design solution and experiencing the physical interaction herself, she's able to immediately iterate on the design. The first thing she notices is how far she needs to tilt the device in order to see the 3D avatar model's perspective shift. She quickly settles on a comfortable degree of tilting by demonstrating the desired range-of-motion and interpolating her animation between these two orientations. Still, she feels the behavior does not feel quite right, so she decides to try swiveling her tablet on its kickstand instead. She notices that the real-time display of gyroscope values in the Sensor Mapper's control panel responds to this motion as she mimics the desired movement, and so she clicks on the "Use gyroscope motion" option along the left-right (Y) axis to try this out. Now it "feels right" and she proceeds with this design.

***Add Additional Viewers on Nearby Devices.*** Chichima next wants to experiment with being able to share her 3D avatar model designs during a video call. At this point she realizes she needs another device so her tablet can host the shared 3D avatar model content, but her vertical desktop monitor can show the video-feed of the other participants in the call. So she opens up a second Viewer on her vertical desktop monitor, where her video-camera is also situated, which lets her quickly try out the same 3D model video, responding to the same sensor data, but on an entirely separate device. From this, she experiments with fading in and fading out her 3D avatar model on top of a backdrop she finds on the web for the Teams videoconferencing app, showing the video-feeds of the hypothetical call's other participants.

***Viewers on Remote Cloud-Connected Devices.*** Chichima next recruits a remote-work colleague so they can try running a shared Escapement session across multiple local and remote devices. Each starts a Viewer on their devices, and now her colleague sees the 3D avatar model shared into and out of the mocked-up video call as Chichma swivels her tablet back and forth. And now Chichima has a brainstorm: swiveling her device left and right to rotate her 3D avatar model in this way makes her realize that she could extend this motion to "turn" the 3D model content *into* or *out of* the call, to make sharing content – or *un-sharing* content – super lightweight and easy. This feels amazing and she even tries a further refinement, where she can rotate her 3D model back and forth when that is the focus of conversation – but then swivel *further left* to instead foreground the video-feed of the other participants; or swivel further right to focus the audience on her video-feed as the presenter. So now sharing content, attending to the audience, or centering her own video feed are all part of one smooth continuous gesture.

***Usage Scenario Summary and Perspective.*** These may or may not be great ideas, but note how Chichima's gone from playing with a small and abstract idea, to experiencing an interactive prototype where she can reflect-in-action and choose her next step, possibly including testing her design with usability participants or stakeholders. And over the course of an afternoon, with each iteration taking just a few minutes to try out, Chichima has experimented with multiple possible interactions, using design materials readily at hand, and all while quickly developing an embodied, visceral understanding of her proposed designs.

## 5 DESIGN STRATEGIES SUPPORTED BY VIDEO-ESCAPEMENT PROTOTYPING

Video-escapement prototyping is a design method that allows designers to operationalize time as a design material, and allows designers to manipulate the temporal dimension of video clips through sensor inputs. This method is separate from — but co-evolved with — the Escapement tool we designed, iterated on, and built (as detailed in Section 3 above). Escapement has been used in scores of extended iterative design sessions and three hackathons, as well as for exploring a half dozen design spaces and focused project areas (some published, some unpublished) since 2017. Over 20 researchers (including the authors), colleagues, students, extended collaborators, and hackathon participants have built multiple interactive prototypes to demonstrate techniques, concepts, or other interaction ideas with the tool. As various sets of designers have worked with the tool over several years, we have observed recurring strategies and patterns-of-use for prototyping sensor-driven animations (whether single device or multi-screen/multi-device) that manipulate video in response to gestures or device motion. These strategies, depicted in Figure 4, show some examples of *demonstrations* [32],

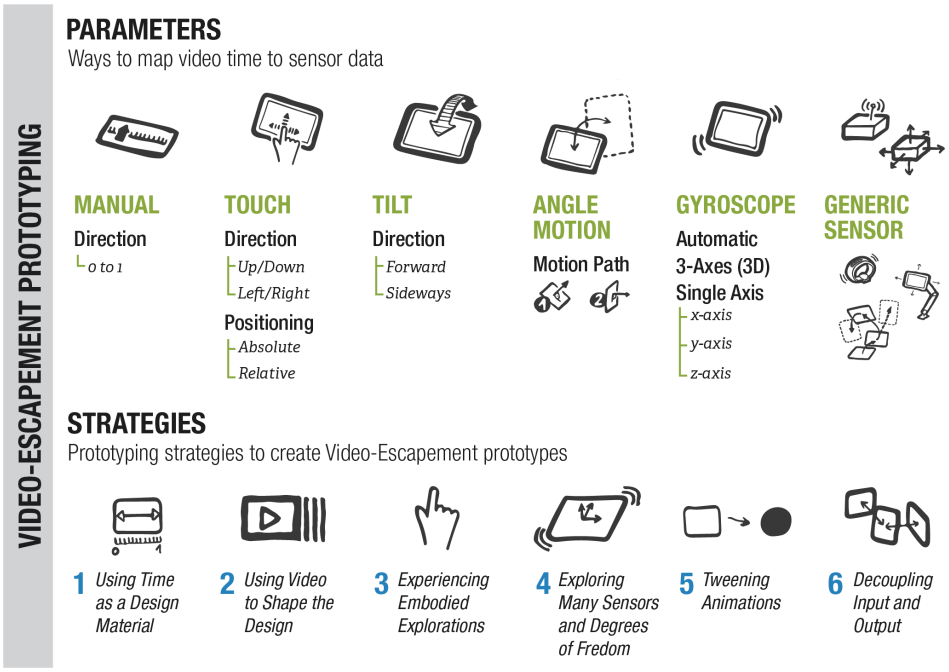


**Figure 4: Top: Escapement provides designers access to various parameters of control, which allows for the manipulation of video with time abstracted out. Bottom: Throughout the years the tool has been in use, we identified prototyping strategies in use by designers as they create video-escapement prototypes.**

where we articulate the *paths of least resistance* [46] provided by our tool.

***Using Time as a Design Material.*** By first pulling time out of the videos, and then providing tools to manipulate the abstracted time dimension, Escapement enables designers to engage with time as a first class design material. For example, Escapement allows designers to easily time-reverse a given video sequence. This offers key functionality because created or recorded videos can be quickly remapped to different sensors or different degrees-of-freedom – while maintaining stimulus-response compatibility – simply by reversing the video playback. This also contributes to Escapement's bricolage-style use of "found video" because such videos are difficult to time-reverse with standard video editing tools, and yet might often happen to capture the opposite of the designer's intended transition or feedback. By providing easy access to the time dimensions through a "reverse playback" radio button (see Figure 3), Escapement lowers the threshold for designers thinking about the way they can manipulate time.

Designers are also able to "pause" time. While Escapement currently offers few features geared towards discrete state transitions, one exception is a Viewer option to start or stop animations in response to finger-down or finger-up events. For example, this lets the designer hold the screen (while moving the device) to "clutch" the current animation state, and then have it resume (start tracking device motion again) on finger-up. Likewise, it can be employed to stop and hold animations once the device has been moved to a desired position or orientation. Another example is to use touch as a toggle to mock up button-press functionality; tapping the screen then triggers playback of the rest of the video animation (this is the one exception where Escapement automatically plays back video over time). This functionality highlights another mechanism by which designers access "time" as a design material, a unique feature of video-escapement prototyping.

***Rapidly Exploring Different Sensors and Degrees-of-freedom.*** As part of the iterative creative exploration process, designers may want to re-use a single animation, but try mapping it to different sensor inputs, or different degrees of freedom. Designers can experiment with design questions, such as whether direct touch, indirect (relative) mouse input, or embodied/tangible tilting or other device movement feel the most appropriate to a postulated interaction. For example, exploring the different experiences enabled by a forward-backward tilt *vs.* a side-to-side tilt involves a simple radio button click, enabling extremely rapid "sensor swapping." Escapement's control panel provides a variety of options, checkboxes, and sliders to quickly swap, change or otherwise experiment with these options, so that the designer can reflect-in-action not only on the "look" of the user interface, but also on the embodied feel (including stimulus-response compatibility, as well as more whimsical aspects of the interaction's "personality").

Sensor data can also be mapped to separate animations across any number of viewers. For example, a user tilting the device 10 degrees might display the first half of an animation in one viewport, and tilting the device an additional 10 degrees may display a separate animation in an entirely different viewport, such as on another device to display a "file transfer". Together, the ability to quickly

try out an animation in response to real sensor data enables highly contextual exploration.

***Engaging in Embodied Exploration.*** The flip side of this is also true: as designers physically manipulate the devices, even in straightforward ways, they learn-by doing [11, 12, 30]. There is power in this approach. By starting out with "simple" ideas, *and actually trying them out interactively*, the designer may become attuned to less-obvious possibilities that tend to suggest themselves in the course of exploration, design iteration, and reflection-in-action [53] on the artifacts produced. For example, a user may tilt the device down to provide a more comfortable writing posture. Yet once enacted through video-prototyping, other ways to leverage this "natural" action (such as automatically revealing pen colors and mode controls) seem more likely to arise; and indeed several small ideas explored in this way often came together into demonstrations that, taken together, seemed far from obvious at the outset, such as multiple simultaneous contextual adaptions appropriate for transitions between sensed reading vs. writing postures of a tiltable display [51]. By supporting embodied exploration, designers seem empowered to try out a variety of choices, including various such contextual, background, or peripheral actions that trigger context-relevant experiences or transitions between different contexts of use.

***'Conversing' with Creative Media.*** Because Escapement lowers the barrier to using and manipulating video content, designers tend to develop a sensitivity or awareness of how the video content could suggest certain interaction designs. For example, designers frequently chose videos that showed changes in either the spatial dimension (such as rotating the view of a 3D map up and down), or the temporal dimension (such as seeing a piece of artwork evolve forward and backward in time). These spatio-temporal changes were frequently paired with physical manipulation such as tilt [51], whereas video content that is less spatial in nature was often paired with a touch gesture. In other words, the tool wasn't simply used as "diverse ways to scrub through video"; rather, we noticed that the content of the videos frequently informed the types of interactions that designers felt were appropriate in an ongoing "conversation" with the video-based design materials [53]. Specifically, designers using Escapement display sensitivity to stimulus-response compatibility by, for example, using a left-right tilt motion to match a horizontal optical flow in a video, and an up-down tilt motion to match video content emphasizing vertical flow (or perhaps opting to mismatch them as a playful, even provocative, design choice). While our system can ingest MP4 video files without regard to length, short video snippets (typically under about 30 seconds in length) tend to be most appropriate for exploring a continuous, animated transition between interface states, if only because of the limited range, input resolution, or susceptibility to noise typical of many proximity, motion, and other sensors.

***Using the Animation Technique of Tweening.*** Escapement can be understood as a digital flipbook that maps input from various sensor sources to a sequence of images *between* two interface states. Animated transitions in presentation tools, such as the Morph transition effect in PowerPoint, offer a convenient way to export short videos suitable for importing into Escapement. The first slide is the first keyframe, and the second slide is edited to produce the second keyframe; the resulting morph animation smoothly and automatically interpolates between the two slides ("keyframes") in a smooth video transition. We have used this capability to quickly generate many interactive prototypes, such as unpublished demonstrations of how applications can respond in richer ways to automatic screen rotation (see accompanying video). Additionally, because designers have access to the generated still image sequences, the animation can be tweaked manually by simply deleting particular image frames, or by adding (or duplicating) others. This is analogous to the way physical filmstrips can be spliced or cut, to add or remove a sub-section of footage. Such direct access to the animation flipbook frames empowered designers to manipulate the final clips directly as they iterated on an interaction design.

***Conceptually Decoupling Input and Output.*** Because multiple Viewers can be open on multiple devices, designers are able to separate different aspects of app functionality across multiple devices or viewports (for example, consider a remote video-conferencing application, with the chat box, system controls, and reaction options displayed on one device, while the streaming video feed appears on another device). Conceptualizing not only such a design distributed across multiple devices ("logical distribution" [3]), but additionally designing graceful transitions in and out of different combinations of devices – and different forms of "distribution" – is a particular challenge for which Escapement is especially well-suited.

The key to supporting designers working on this type of design is the conceptual decoupling of input and output. By abstracting time out of the interaction, the replayed application behaviour is no longer tied to a specific device, but can instead be triggered by a variety of supported sensors, or selected DoF's thereof. The same sensor stream can be interpreted differently in different viewports, running either on the same device, an externally connected monitor, or an entirely separate device. An additional advantage of such input/output decoupling is the way the tool requires designers to consider what will play on any connected viewport, or on audience vs. presenter views in the context of remote work. By being forced to consider the content in the viewport separately from the sensor data triggering the content, designers can expand the landscape of devices and device ecologies they are designing for.

## 6 ESCAPEMENT DESIGN ITERATION AND EVOLUTION

Some version of Escapement has been in active use by both novice and expert designers since 2017, and the functionality of the tool itself has evolved and expanded in that time. Early versions of Escapement were used in diverse scenarios from one-off rapid explorations, to Hackathon brainstorming, to polished, high-fidelity research prototypes [43, 51] by more than 20 researchers and designers (including the authors). These incarnations of the tool embody our evolving understanding of video-based prototyping, which we describe in this section as part of our research-through-*(tool)*-design process [59].

### 6.1 Version Zero: Early one-off demos

Escapement had its origins in one-off demos, such as an opening/closing animation for a mocked-up dual-screen device (see

accompanying video); our primary motivation at the time was to explore continuous, analog responses to device movements and changes in device posture. Instead of implementing a complex animation, we used "found" video of a sunrise time-lapse, composited beneath a design-tool-generated render of home screen icons swinging into place. Frames of this video were then shown according to the hinge angle between the two screens. While that demo proved to be a one-off, the potential utility of coupling short video snippets with interactive sensor animations captured our imagination, and we slowly started adopting this code fragment to prototype sensor-mediated interactions in other design projects.

## 6.2 Version 1: Single device, single sensor

We next built an application, dubbed *TiltVideoPlayer* at the time, to explore and interactively prototype techniques for novel tilt responses of adjustable displays [51]. The TiltVideoPlayer allowed a user to play back video files interactively, in response to a one-degree-of-freedom tilting motion, with the angle of the device mapped to the frames of the recorded video. This allowed interactive non-sequential access, forward or backwards, as the user adjusted the display up or down. The designers working with TiltVideoPlayer used screen recordings of various applications as well as stock time-lapse videos (sometimes with rendered UI elements added on top) to quickly iterate on designs prior to implementing the most compelling ideas in code. The designers found TiltVideoPlayer to be a powerful prototyping tool: enabling rapid exploration to quickly try out ideas, engage in reflection-in-action [53] or knowing-through-action [11, 12] with functional prototypes, and evaluation with end-users.

While one core aspect of Escapement – controlling video playback via sensor data – was already present in this early design, critically it lacked the key concept of "normalization" of both time and input dimensions to $0 \rightarrow 1$. So, for example, care had to be taken to craft a video of the correct length and with the "right" transfer function (linear, ease-in/ease-out, etc.) "baked in" at the time of capturing or exporting the video, making it tedious to tweak the feel and personality of the interaction. Other limitations of this TiltVideoPlayer version of the tool included support for only a single DoF of the tilt sensor, as well as a single animation viewport displayed on the same device. Likewise, there was no editable transfer function, no notion of an input dimension, and no support for touch gestures or other motion sensors. Nevertheless, for ideas that could be explored within these limitations, designers were still able to quickly explore, iterate, and viscerally experience – and then either discard or refine – a wide variety of tilt-based interactions.

## 6.3 Version 2: Multiple devices, multiple sensors

The tool then went through a major re-conception and re-development, to the point where 1) designers could replay videos on multiple viewports running on the same or separate devices; and 2) read sensor data from many more sensors, both built-in (such as the gyro and magnetometer) and arbitrary attached sensors (such as a proximity sensor connected via an Arduino). We additionally added more sensor options, including the ability to filter, define cutoffs, choose input DoF's, demonstrate motion paths by example, and so forth.

This version of the tool was used to support the design of in-air, poseable, multi-device assemblages as described in [43], as well as numerous previously-unpublished scenarios for remote work, dual-screen devices, and application responses to screen rotation (see accompanying video). This updated version of the tool allowed designers to quickly and effectively prototype cross-device interactions (interact with a sensor on device A to effect a change on devices A, B, C, D, ...), thereby supporting more complex device ecologies with transitions across multiple devices. The tool also provided additional data filtering (including easing and transfer functions).

During the pandemic, challenges of driving Escapement prototypes on multiple devices that needed to connect across firewalls, institutional and home networks, or unreliable router configurations motivated us to move beyond local UDP multicast and further implement shared state via the cloud (as detailed in Section 3.2.2 above). This includes performant streaming of sensor data and other animation state via the cloud; while network performance depends on home network bandwidth, geographical location, and other factors, one-way latencies as low as 20ms from edge machines outside the cloud, into cloud data-centers, are possible [8]. Our current implementation of this feature has not yet been optimized to minimize round-trip latency, or tested via emerging network advances such as 5G, so additional latency gains are possible. In practice we found it was not unusual to see added latencies well over a hundred ms when using Escapement across the cloud; but this offers an authentic representation of real-time cloud performance with commonplace home setups in North America, for example. And since Escapement's cloud-connectivity feature is optional, designers can still prototype scenarios that require low-latency real-time sensor response by using multiple viewports on the same device, or with lower network latency across device settings where UDP multicast is available.

## 6.4 Summary of Tool Evolution.

While the core concept of controlling video playback via sensor data persisted throughout each version of the tool, other features continued to be refined and extended as we developed our concept of video-escapement prototyping. This long-term evolution of Escapement helped us identify the core *relevant* [59] design features, including: 1) using time as a design material, 2) manipulating video in a conversational and 3) embodied way, 4) rapidly exploring different sensors and degrees-of-freedom, 5) using animation techniques like tweening, and 6) conceptually decoupling input from output (see Figure 4).

This longitudinal design and usage helped us to validate the effectiveness of the tool [32] beyond usability evaluations [15, 47]. The Design Strategies section (Section 5) was distilled from long-term *observed usage* [32], and captures commonly observed techniques for developing interactions with the tool. These techniques have been used by designers over the tool's lifetime, highlighting the unique capabilities, *threshold* (ease in getting started), and *ceiling* (ability to generate expressive results) [46]. In addition to the observed usage, our *evaluation by demonstration* [32] (Section 4) highlights both *what the tool might support*, as well as *how users*

*might work with it.* Together, these observations provide confidence in the *relevance* and *extensibility* [59] of the tool.

# 7 DISCUSSION

## 7.1 Supporting Practitioner Design Values

Because video prototypes are already widely embraced by interaction designers, augmenting such video prototypes to be *interactive* is an important way to increase expressive leverage (enabling designers to achieve more with less) and expressive match (close alignment between tool output and designer goal) [47]. Additional creative practitioner values [48] we support include dynamic responsiveness, quick-start (the ability to rapidly begin engaging in design activities) and experimentation (extremely tight loops between making a change to a system and seeing the resulting change). Escapement is an example of a tool that supports *design by enaction* [37]. Specifically, our tool allows designers, developers, and users to have an embodied experience with a particular design in context at an extremely early stage of the design process. Whether a designer wants to start their ideation process by considering what physical interactions are a good fit, or has a specific application behaviour in mind, the tool supports multiple ideation paths. However, while the tool is particularly strong at providing access to sensor data and collections of devices, the types of interactions it enables may not be appropriate for all experiences. For example, it may not be appropriate to design an email client that requires users to tilt their devices in order to send a message, or require gross motor actions for a behavior that requires fine motion control. On the other hand, the ability to quickly try out ostensibly "bad" ideas may lead to surprises, or spur the next "better" idea; or at the very least offer a fast way to disabuse designers (and perhaps even executives) of dubious design choices.

## 7.2 Design at Scale

Escapement allows designers to experiment with the same interaction across different layers of scale. Simply by running the Viewer on different devices, the designer can explore the same animation on a single mobile device, or multiple connected tablets, or a large desktop monitor, or even large wall displays or electronic whiteboards, increasing both in number of devices and screen real estate. This flexibility – an important aspect of a novel design system [47] – is supported by the conceptual decoupling of input and output, and allows designers to rapidly scale their experimentation up across diverse device ecologies. For example, one could be designing a notification system, first trying different sensors or degrees of freedom to trigger the notification on a single device. Then switch to e.g., a dual screen device, or desktop notification. The same video can be repurposed and experienced for each of these use cases, allowing for extremely rapid, highly contextual, embodied exploration with the particular sensors, input resources, and display dimensions manifest on each form-factor.

## 7.3 Continuous vs. Discrete Interaction and State Transitions

Escapement is well-suited to continuous real-time sensor responses, as it had its genesis in exploring such interaction techniques and scenarios. By contrast, so-called WIMP interfaces are characterized by dichotomous modes and discrete state transitions, yet much of everyday life is analog and continuous; it seems there remains much potential in exploring and bringing such nuances to digital experiences through continuous, analog, sensor-mediated interaction techniques. This focus means that the Escapement tool was not designed for discrete events that transition from one state to another, such as click-throughs of multiple application screens. Having said this, other than the risk of making the tool over-complex, there is no fundamental design or technical reason that Escapement could not be extended with support for state transitions in the future.

## 7.4 Video as a Raw Material

When working with video as a raw material in the context of Escapement, it is important to note that while an arbitrary video can be used, the video contents should be relevant to the task at hand. In the most basic scenario, a designer might be testing a video playback tool that responds to tilt, for which any video can get the job done. Going beyond tilt-responsiveness, the designer might want to explore, for instance, scrolling through a panoramic photo as a function of tilt. In that case, the video should be more specific to the panoramic-photo-viewing task at hand. However, because a wide range of video content can be imported and played by Escapement, this also opens up opportunities for serendipity, either by bringing a video to life as a quick experiment, or by having a target video that a designer tries out with different types of input. As described in Section 3, Escapement is not designed for one-shot animations that play once an action has taken place - rather the video responds *while* the action is happening.

## 7.5 Limitations

While future work should further compare and contrast video-escapement prototyping to other prototyping tools, our current objective in this paper is to articulate the benefits of a new tool with new capabilities, affordances, and points of friction. Like any other tool Escapement offers strengths and weaknesses that are appropriate for some design tasks (sensor-mediated, continuous response to user-in-the-loop input, cross-device interfaces) but not others (such as scenarios requiring clicking through of a series of discrete states).

Perhaps a more fundamental limitation of Escapement is its focus on rapid-prototyping of *applications that use screens*, providing feedback solely via the visual channel. At present, it is difficult to see how to adapt Escapement's core conceit of non-linear, sensor-mediated playback to other feedback modalities such as auditory, vibrotactile, or actuated haptic response; but perhaps concepts from robotics, control systems, or other signal processing techniques could be brought to bear. Lacking that, even one-shot triggering of auditory cues, for example, would be a welcome addition to the tool.

Another inherent quality of Escapement is its abstraction of sensors to one-dimensional data streams that can be interpreted as (or projected onto) a single $0 \rightarrow 1$ dimension. For example, a designer cannot use Escapement as currently conceived to build a real-time interactive prototype of an X-Y panning gesture based on an (arbitrarily moving, two-dimensional) touch gesture. As another

example, one could not prototype a Virtual Reality (VR) experience with a free-moving viewpoint tracked by a multidimensional head-tracker. However, for the purposes of video-escapement prototyping, we have also observed that designers will sometimes realize clever work-arounds sufficient to convey their ideas, such as simulating pinch-to-zoom gestures by tracking single-touch motion of just one of the two fingers involved.

The current lack of a path from an Escapement prototype to running code is another, perhaps more subtle, limitation of our approach at present. Video-escapement prototypes are great for experimenting with sensors, transfer functions, and the corresponding graphical response, but once the designer settles on a satisfying combination the tool currently offers no way to persist this state or emit code that could be re-used in application development. Likewise, some (even limited) support for scripting, or emitting design iterations to a Jupyter-like notebook, could be valuable.

Further, there is a risk that the low-friction path to no-code interactive prototypes afforded by Escapement may trap designers at the "rapid prototype" stage, making it difficult to transition their ideas to more fully fleshed-out running applications. As one way to partially address this, we are considering encapsulation of Escapement's core sensor filters and easing functions as a library that could be imported into a software developer's code base, along with persisted settings and parameter values.

## 8 CONCLUSION AND FUTURE WORK

In this paper, we have introduced the notion of video-escapement prototyping, a design technique that allows designers to manipulate the abstracted time dimension of video clips via streamed sensor data. We also introduced Escapement, a novel video prototyping tool that embodies this notion, and has allowed designers to rapidly explore and viscerally experience sensor-mediated interactions across diverse ecologies of devices, screens, input modalities, and viewers. We have also described the way experienced and novice designers have used Escapement over the last several years, and especially how video-escapement prototyping influences their design activities and final output.

While the current version of the tool focuses on continuous, video-based interactions, the notion of "escapement prototyping" has the potential to enable prototyping in other modalities. While a full exploration is outside the scope of this paper, future iterations of the tool that go beyond the visual modality could continue to expand the design space. Better support for alpha-transparency or chroma-key style transparency that would allow compositing an Escapement demo on top of another live, running application or live camera feeds (e.g. a remote meeting) would enable additional prototyping opportunities. Also part of future work — and with a return to in-person studies and workshops hopefully feasible in the not-too-distant future — we are keen to further develop and observe interaction designers' use of Escapement to more fully establish its design properties as a creativity support tool.

With Escapement, we envision a future where video-escapement prototyping may become more widely adopted into interaction design practice as well as integrated into a variety of design tools. This would empower a wider population than ever before to rapidly explore, ideate, and refine continuous interaction techniques across a range of design materials and user scenarios, greatly accelerating HCI research, design, and practice across both well-known and emerging inputs and sensors.

## ACKNOWLEDGMENTS

Thanks to the anonymous reviewers for their thoughtful, thorough, encouraging, and kind feedback. Thanks to Gierad Laput, whose early explorations were instrumental in exploring the concept in its most nascent form.

## REFERENCES

[1] Ronald Michael Baecker. 1969. *Interactive computer-mediated animation.* Technical Report. MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC.
[2] Krishna Bharat and Luca Cardelli. 1996. Migratory applications. In *International Workshop on Mobile Object Systems*. Springer, 131–148.
[3] 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
[4] Nestor Burtnyk and Marceli Wein. 1976. Interactive skeleton techniques for enhancing motion dynamics in key frame animation. *Commun. ACM* 19, 10 (1976), 564–569.
[5] Bill Buxton. 2010. *Sketching user experiences: getting the design right and the right design.* Morgan kaufmann.
[6] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1€ filter: a simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2527–2530.
[7] Badrish Chandramouli, Guna Prasaad, Donald Kossmann, Justin Levandoski, James Hunter, and Mike Barnett. 2018. Faster: A concurrent key-value store with in-place updates. In *Proceedings of the 2018 International Conference on Management of Data*. 275–290.
[8] Batyr Charyyev, Engin Arslan, and Mehmet Hadi Gunes. 2020. Latency comparison of cloud datacenters and edge servers. In *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 1–6.
[9] Alan Cooper, Robert Reimann, David Cronin, and Christopher Noessel. 2014. *About face: the essentials of interaction design.* John Wiley & Sons.
[10] Nigel Cross. 2011. *Design thinking: Understanding how designers think and work.* Berg.
[11] Peter Dalsgaard. 2014. Pragmatism and design thinking. *International Journal of Design* 8, 1 (2014), 143–155.
[12] Peter Dalsgaard. 2017. Instruments of inquiry: Understanding the nature and role of tools in design. *International Journal of Design* 11, 1 (2017).
[13] Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowitcz, Derek Nowrouzezahrai, Ravin Balakrishnan, and Karan Singh. 2008. Video browsing by direct manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 237–246.
[14] George Fitzmaurice and William Buxton. 1994. The chameleon: Spatially aware palmtop computers. In *Conference companion on Human factors in computing systems*. 451–452.
[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*. 111–120.
[16] Saul Greenberg, Sheelagh Carpendale, Nicolai Marquardt, and Bill Buxton. 2011. *Sketching user experiences: The workbook.* Elsevier.
[17] Donatien Grolaux, Peter Van Roy, and Jean Vanderdonckt. 2004. Migratable user interfaces: beyond migratory interfaces. In *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004.* IEEE, IEEE, USA, 422–430.
[18] 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 (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2773–2782. https://doi.org/10.1145/2556288.2557170
[19] Björn Hartmann. 2009. *Gaining design insight through interaction prototyping tools.* Stanford University Stanford, CA.
[20] Björn Hartmann, Leith Abdulla, Manas Mittal, and Scott R Klemmer. 2007. Authoring sensor-based interactions by demonstration with direct manipulation and pattern recognition. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 145–154.
[21] James Hollan, Edwin Hutchins, and David Kirsh. 2000. Distributed Cognition: Toward a New Foundation for Human-Computer Interaction Research. *ACM Transactions on Computer-Human Interaction* 7, 2 (jun 2000), 174–196. https://doi.org/10.1145/353485.353487

[22] Stephanie Houde and Charles Hill. 1997. What do prototypes prototype? In *Handbook of human-computer interaction*. Elsevier, 367–381.

[23] Scott E Hudson and Jennifer Mankoff. 2006. Rapid construction of functioning physical interfaces from cardboard, thumbtacks, tin foil and masking tape. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*. 289–298.

[24] Edwin Hutchins. 1995. *Cognition in the Wild*. MIT press.

[25] Tero Jokela, Jarno Ojala, Guido Grassel, Petri Piippo, and Thomas Olsson. 2015. A Comparison of Methods to Move Visual Objects Between Personal Mobile Devices in Different Contexts of Use. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark) *(MobileHCI '15)*. ACM, New York, NY, USA, 172–181. https://doi.org/10.1145/2785830.2785841

[26] James C Kaufman. 2016. *Creativity 101*. Springer publishing company.

[27] Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, and George Fitzmaurice. 2014. Kitty: sketching dynamic and interactive illustrations. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. 395–405.

[28] John F Kelley. 1983. An empirical methodology for writing user-friendly natural language computer applications. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. 193–196.

[29] Iyad Khaddam, Jean Vanderdonckt, Salah Dowaji, and Donatien Grolaux. 2020. Towards Rapid Prototyping of Foldable Graphical User Interfaces with Flecto. *Proceedings of the ACM on Human-Computer Interaction* 4, ISS (2020), 1–33.

[30] Scott R Klemmer, Björn Hartmann, and Leila Takayama. 2006. How bodies matter: five themes for interaction design. In *Proceedings of the 6th conference on Designing Interactive systems*. 140–149.

[31] Clemens N Klokmose, James Eagan, Siemen Baader, Wendy Mackay, and Michel Beaudouin-Lafon. 2016. Webstrates: demonstrating the potential of Shareable Dynamic Media. In *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*. Association for Computing Machinery, New York, NY, USA, 61–64.

[32] 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*. 1–17.

[33] David Ledo, Jo Vermeulen, Sheelagh Carpendale, Saul Greenberg, Lora Oehlberg, and Sebastian Boring. 2019. Astral: Prototyping Mobile and Smart Object Interactive Behaviours Using Familiar Applications. In *Proceedings of the 2019 on Designing Interactive Systems Conference*. 711–724.

[34] David Ledo Maira. 2020. Designing Interactive Behaviours for Smart Objects. (2020).

[35] Germán Leiva and Michel Beaudouin-Lafon. 2018. Montage: A video prototyping system to reduce re-shooting and increase re-usability. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 675–682.

[36] Germán Leiva, Jens Emil Grønbæk, Clemens Nylandsted Klokmose, Cuong Nguyen, Rubaiat Habib Kazi, and Paul Asente. 2021. Rapido: Prototyping Interactive AR Experiences through Programming by Demonstration. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. 626–637.

[37] Germán Leiva, Nolwenn Maudet, Wendy Mackay, and Michel Beaudouin-Lafon. 2019. Enact: Reducing designer–developer breakdowns when prototyping custom interactions. *ACM Transactions on Computer-Human Interaction (TOCHI)* 26, 3 (2019), 1–48.

[38] Yang Li and James A Landay. 2005. Informal prototyping of continuous graphical interactions by demonstration. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*. 221–230.

[39] Youn-Kyung Lim, Erik Stolterman, and Josh Tenenberg. 2008. The anatomy of prototypes: Prototypes as filters, prototypes as manifestations of design ideas. *ACM Transactions on Computer-Human Interaction (TOCHI)* 15, 2 (2008), 1–27.

[40] Panagiotis Louridas. 1999. Design as bricolage: anthropology meets design thinking. *Design Studies* 20, 6 (1999), 517–535.

[41] Wendy E Mackay. 1988. Video Prototyping: a technique for developing hypermedia systems. In *CHI'88 Conference Companion Human Factors in Computing Systems*, Vol. 5. Citeseer, 1–3.

[42] Wendy E Mackay. 2002. Using video to support interaction design. *DVD Tutorial, CHI* 2, 5 (2002).

[43] Nicolai Marquardt, Nathalie Henry Riche, Christian Holz, Hugo Romat, Michel Pahud, Frederik Brudy, David Ledo, Chunjong Park, Molly Jane Nicholas, Teddy Seyed, et al. 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*. 1252–1268.

[44] Judith Martens, Thomas Franke, Nadine Rauh, and Josef F Krems. 2018. Effects of low-range latency on performance and perception in a virtual, unstable second-order control task. *Quality and User Experience* 3, 1 (2018), 1–17.

[45] Michael McCurdy, Christopher Connors, Guy Pyrzak, Bob Kanefsky, and Alonso Vera. 2006. Breaking the fidelity barrier: an examination of our current characterization of prototypes and an example of a mixed-fidelity success. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*. 1233–1242.

[46] Brad Myers, Scott E Hudson, and Randy Pausch. 2000. Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI)* 7, 1 (2000), 3–28.

[47] Dan R Olsen Jr. 2007. Evaluating user interface systems research. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*. 251–258.

[48] Srishti Palani, David Ledo, George Fitzmaurice, and Fraser Anderson. 2022. "I don't want to feel like I'm working in a 1960s factory": The Practitioner Perspective on Creativity Support Tool Adoption. In *ACM CHI Conference on Human Factors in Computing Systems*. 275–290.

[49] Taylor Palmer and Jordan Bowman of UX Tools. [n. d.]. 2020 tools survey results. Retrieved October 26, 2021, from https://uxtools.co/survey-2020.

[50] Robert Penner. 2002. *Robert Penner's Programming Macromedia Flash MX*. McGraw-Hill, Inc.

[51] Hugo Romat, Christopher Collins, Nathalie Henry Riche, Michel Pahud, Christian Holz, Adam Riddle, Bill Buxton, and Ken Hinckley. 2020. Tilt-Responsive Techniques for Digital Drawing Boards. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*. Association for Computing Machinery, New York, NY, USA, 500–515. https://doi.org/10.1145/3379337.3415861

[52] Ibrahim Sabek, Badrish Chandramouli, and Umar Farooq Minhas. 2019. CRA: Enabling data-intensive applications in containerized environments. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 1762–1765.

[53] Donald A Schon. 1979. The reflective practitioner. *New York* (1979).

[54] Ben Shneiderman. 2007. Creativity Support Tools: Accelerating Discovery and Innovation. *Commun. ACM* 50, 12 (Dec. 2007), 20–32. https://doi.org/10.1145/1323688.1323689

[55] Erik Stolterman. 2008. The nature of design practice and implications for interaction design research. *International Journal of Design* 2, 1 (2008).

[56] Anna Vallgårda and Ylva Fernaeus. 2015. Interaction design as a bricolage practice. In *Proceedings of the ninth international conference on tangible, embedded, and embodied interaction*. 173–180.

[57] Laurie Vertelney. 1989. Using video to prototype user interfaces. *ACM SIGCHI Bulletin* 21, 2 (1989), 57–61.

[58] Jishuo Yang and Daniel Wigdor. 2014. Panelrama: enabling easy specification of cross-device web applications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2783–2792. https://doi.org/10.1145/2556288.2557199

[59] John Zimmerman, Jodi Forlizzi, and Shelley Evenson. 2007. Research through design as a method for interaction design research in HCI. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 493–502.