Analyzing Barehand Input Mappings for Video Timeline Control and Object Pointing on Smart TVs

by

Futian Zhang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2021

Author's Declaration

Statement of Contribution

This thesis includes first-authored peer-reviewed material that has appeared in conference proceedings published by Springer. The conference paper from which I have adapted content is the following:

- Futian Zhang, Sachi Mizobuchi, Wei Zhou, Taslim Arefin Khan, Wei Li, and Edward Lank. Leveraging CD Gain for Precise Barehand Video Timeline Browsing on Smart Displays. In *Proceedings of the 18th IFIP TC13 International Conference on Human-Computer Interaction* (INTERACT '21). To Appear.

I understand that my thesis may be made electronically available to the public.

Abstract

Smart TVs are getting popular in recent few years. Given the emerging feature of distant bare hand control, one challenge is how to perform common tasks with this new input modality. Two tasks are discussed in this thesis including video timeline control task and object pointing task.

For video timeline control task, we explore CD gain functions to support seeking and scrubbing tasks. We demonstrate that a linear CD gain function performs better comparing with either a constant function or generalised logistic function (GLF). In particular, Linear gain is faster than a GLF and has lower error rate than Constant gain. Furthermore, Linear and GLF gains' average temporal error when targeting a one second interval on a two hour timeline (+/- 5s) is less than one third the error of a Constant gain.

For object pointing task, we design five select strategies to compare the performance, including one Positional based mapping, one Rate based mapping, one Positional + Rate based mapping and two Traditional TV remote style mappings. We picked the first three techniques for our user study. Through a series of experiments, we demonstrate that Positional mapping is faster than other mappings when the target is visible but requires many clutches in large targeting spaces. Rate-based mapping is, in contrast, preferred by participants due to its perceived lower effort, despite being slightly harder to learn initially. Tradeoffs in the design of target selection in smart tv displays are discussed.

Acknowledgements

I would like to thank all the little people who made this thesis possible.

First, I would thank my supervisor, Prof. Edward Lank, who gives me support and guidance during my graduate study. He respects my decision and puts his students first before himself. I'm so lucky to be one of his students.

My sincere thanks also go to Prof. Jian Zhao and Prof. Keiko Katsuragawa at University of Waterloo for being my thesis readers and providing valuable feedback. It would not have been possible without their input that my thesis could be in such a great shape.

I would also like to thank my collaborators, Keiko Katsuragawa, Sachi Mizobuchi, Wei Zhou, Taslim Arefin Khan, Wei Li, Hemant Surale, Quentin Roy, Daniel Vogel, and also colleagues at Huawei Canada. You help me become a better researcher and it's my pleasure to work with all of you.

I also want to thank my friends in the Human-Computer Interaction Lab at University of Waterloo, Damien, Rina, Jay, Constant, Bahar, Anastasia, Hanae, Nabil, Shawon, Hemant, Tim, Sangho, Johann, Nikhita, Matthew, Jeremy, Allen, Antony, Graeme, Katherine, Ludwig, Margaret and Nalin; and to my friends at Waterloo: Jerry, Lucas, Peng, He, Yuming, Xiaoyang, Guanlin. It's nice to have all of you in my life.

I would like to convey my thankfulness to ZJU alumni association at Waterloo. People there help me a lot like family when I arrived at Waterloo. That means a lot to a student living abroad.

I'd also like to thank Lynette for being my friend who attended my thesis presentation and captured precious moments that I forgot to record. And of course for sharing those cute cat videos with me.

Finally, I wish to dedicate this thesis to my dearest parents for their endless love and support.

Dedication

To Mom and Dad.

Table of Contents

List of Figures

Chapter 1

Introduction

Many modern televisions are considered *Smart TVs*, primarily due to the addition of wireless internet (WiFi) connectivity and a set of apps to connect to a variety of on-line services. One of the primary use-cases of these Smart TVs is to consume video, typically through access to various video streaming services such as those offered by Netflix, Amazon, Disney, HBO, Hulu, and others. To complete the video watching tasks, users have to browse and select the video, and then start watching the video. While a basic use-case for streamed video would be to watch the video stream from start to finish, there are also occasions where users wish to navigate within a video stream using video timelines to rewatch a portion, to return to a location due to interrupted viewing, or even to skip forward through advertisements, credits, or uninteresting scenes.

One feature that has been recently added to Smart TVs is the ability to control the display at a distance using barehand gestures (see, for example, Huawei's X65, Hisense U7, and Samsung F-series displays). Users of these displays typically hold their hand up in front of their body and perform gestures and movements to control the display. Using optical or infrared images, cameras track the users hand position and recognize gestures and movement to support barehand input. Given the fact that barehand gestures were introduced to many common interactions in Smart TVs, we would argue that the ultimate goal of barehand gesture input to SmartTVs is to dispense entirely with the remote control. Given the frequency with which remotes are misplaced and the plethora of remotes that a typical household contains the convenience of barehand gestural control is clear.

We acknowledge that there are many challenges associated with barehand gestural control of SmartTVs, including issues of tracking algorithms, gesture design, and turn taking [38], to name only a few. This thesis focuses specifically on two interactive challenges, supporting video timeline control and object pointing, and evaluates different CG gain functions and object mapping strategies in smart display control.

1.1 Video Timeline Control

Video timeline control comprises two distinct tasks: seeking and scrubbing. The seeking task involves acquiring a specific location along the timeline, typically a specific elapsed time within the video. The scrubbing task, in contrast, requires a user to traverse the video in a controlled way, monitoring keyframes, to locate a specific scene whose timing they may not know. In the seeking task, one challenge is that, with limited clutching, we must provide users with sufficient precision to target a specific portion of the video. In a two-hour video (7200 seconds), one-second-level precision would be ideal, but, on a 4K display, second-level precision requires sub-pixel targeting. Conversely, in the scrubbing task, users must be able maintain an intermediate speed such that they can effectively identify and home-in on a desired scene.

Because of the need to support selectable time periods within video with high precision, a variety of researchers have proposed solutions to simplify precise seeking and scrubbing [9, 11, 20, 22, 34]. Despite these innovations, performing seeking and scrubbing tasks on video timelines remains challenging, particularly considering the (frequently limited) input affordances of modern Smart TV remotes.

While one can imagine various pointing enhancements to support the selection of sparse on-screen targets (a bubble cursor, various forms of object pointing, target manipulations [14, 15, 29]), as Balakrishnan [3] notes, cursor or target based pointing facilitation techniques assume significant whitespace on the display. However, input along the timeline is continuous. To map barehand movement onto timeline manipulation, we must either replace the timeline (with attendant disadvantages of changing the fundamental video interaction that users are familiar with), or we must choose an appropriate Control-to-Display Gain (CD Gain) function [31] to effectively support range of movement (from beginning to end of the timeline), precision of movement (to enable second or near-second level selection along the timeline), and control of speed (to support key frame monitoring during scrubbing along the timeline). In this thesis, we explore this question of the impact of well-chosen CD Gain functions on barehand video timeline manipulation.

An open question that we faced was what would constitute a good CD Gain function, and, indeed, whether it was even possible to identify a good CD Gain function given the above conflicting requirements [31]. Naively, we assumed that the literature would identify an appropriate CD gain function for both seeking and scrubbing tasks, but, based on our reading of the literature, this was not the case. There are numerous instances of research that explore the mapping of freehand movement onto cursor movement [7] and significant recent research in selecting CD gain functions [31], but, even after careful exploration of this related literature, we are left with questions, including:

1. What form of CD gain function is most effective for barehand video timeline control? Should it be a constant CD Gain, similar to direct manipulation, or should some form of cursor acceleration/non-constant gain be used?
2. How well do different variants of CD Gain functions work for both seeking and scrubbing tasks? Is there one CD Gain function that can balance the needs of seeking and scrubbing, or must we prioritize one over the other?

To the best of our knowledge, for the domain of barehand video timeline control, it is not possible to glean from the current research literature answers to these questions.

To probe these questions, this thesis presents a controlled experiment that explores CD Gain functions for barehand video timeline control. We design three primary experimental tasks: interval targeting, where a user acquires a specific spatial region on the timeline with a given tolerance i.e. a time interval selection; scene seeking, where a user moves through the timeline with the goal of identifying a specific scene (via keyframes) that they wish to view; and precise time targeting, where a user moves through the timeline to acquire as accurately as possible a specific 1 unit interval (i.e. one second) in an 7200 unit (i.e. two hour) timeline. Overall, we find that pointer acceleration – whether linear or non-linear – significantly outperforms constant CD Gain, and that a Linear CD Gain function presents overall advantages in throughput.

1.2 Object Pointing

Object pointing is one of the key interaction in HCI domain [25, 15, 29, 14]. To explore object pointing via barehand interaction, we analyze the potential input mappings possible for gesture control and identify three primary object-pointing input paradigms for selection: discrete, positional mapping, and rate-based input. Through a pilot study, we highlight that discrete is sub-optimal and so eliminate it from further configuration. Through a follow-on study, we evaluate three different variants of position mapping and rate-based mapping to identify performance characteristics of each.

Our results argue that position and rate-based mappings have complementary benefits. When the input space is simply one screen of the display (e.g. for a video player), then position mapping is highly effective. However, as the input space grows – for example, a video on-demand interface may require extensive scrolling and paging – we observe a strong user preference for rate-based techniques. We conclude this part by highlighting the implications for the design of barehand-controlled object-based pointing techniques for Smart TV Input.

1.3 Contribution

The thesis makes the following contributions:

- Compares three CD gain functions for barehand video timeline control to support seeking and scrubbing tasks.
- Designs five mappings for bare hand object pointing. Compares three of them and provides design guidelines based on the content.

Chapter 2

Related Work

This chapter includes related work on mid-air video timeline control and mid-air object pointing. Both of them involves mid-air pointing interaction. For mid-air video timeline control, previous video timeline control with other modalities and CD gain functions are explored. And object pointing techniques are related to mid-air object pointing as well.

2.1 Video Timeline Control

Even ignoring the problem of barehand video timeline control, the manipulation of video timelines is a recognized problem in HCI research. One common approach to video control is to leverage various forms of content analysis to create sensical representation of the video during forwarding. For example, the SmartPlayer system [9] and Pongnumkul et al.'s [34] content-aware timelines both perform basic analysis of the video scene to modify representations of the video so users can better analyze content during seeking tasks. Alternatively, but still in the domain of content analysis, there exist systems that perform basic analysis of video content so that users can directly manipulate an item in the video image to control video playback to precisely select an individual frame [11, 20, 22, 26].

One alternative approach to controlling video is to replace the video timeline with another structure, for example with hierarchies of keyframes or other structured representations. While a full review of alternative browsing mechanisms is outside the scope of this related work, the interested reader can refer to Schoeffmann et al.'s survey of video manipulation techniques [35].

2.2 CD Gain

In the Windows/Icons/Menus/Pointer (WIMP) paradigm of Graphical User Interface (GUI) input, a CD Gain function maps the movement of a physical input device (the computer mouse, the finger on a touchpad, the hand in the air) onto on-screen pointer movement. CD Gain can either be constant, where speed and/or displacement of the input is multiplied by a fixed scalar value to control on-screen movement, or it can be a dynamic function where the CD Gain function dynamically increases as input speed increases. Constant CD Gain is frequently used in touch-screen based interfaces, primarily because the one-to-one mapping of finger position to on-screen position must be preserved. However, any time there exist spatial offset between the input device and the representation of position, i.e., the on-screen pointer, CD Gain need not be constant. Instead, it can be some form of dynamic function.

The term *pointer acceleration* refers to the use of a dynamic CD Gain function. While commercial systems have included pointer acceleration via dynamic CD Gain functions for at least two decades, Casiez et al. [8] were the first to demonstrate, in 2008, the benefits of dynamic CD Gain for high precision pointing during Fitts's Law tasks in interfaces; prior to this, some thought pointer acceleration could not provide benefits in pointing performance due to the fundamental characteristics of Fitts's Law [19].

Given the seminal work of Casiez et al. in the analysis of the effects of pointing precision, a significant body of recent work exists in analyzing and selecting appropriate dynamic CD Gain functions to support optimal pointer acceleration. One example of this work is work by Casiez and Roussel [7], who introduced a toolkit that allows examination of different real-world pointer acceleration functions by contrasting the movement characteristics of the input device (mouse, touchpad) and the on-screen displacement that results from input device movement. In selection of CD Gain functions, Nancel et al. [31] present an analysis of how best to implement pointer acceleration given the distance needed for pointing and the precision needed for pointing (i.e. function specificiation given the maximum gain needed for distant targeting and the minimum gain needed for sufficient precision).

While the above work on pointer acceleration demonstrates that pointer acceleration aids in Fitts-style targeting tasks and provides guidance on how to identify the minimum and maximum gain necessary for precision and range, respectively, the literature – to the best of our knowledge – is relatively silent on the mid-range of the acceleration function beyond noting the differences in form across operating systems [7]. However, in video timeline control, alongside precision (e.g. targeting to the second across a two hour movie) and range (e.g. targeting the entire range of a two hour timeline), a common task of users is scrubbing, where, in a controlled movement, users traverse the video scanning keyframes for desired scenes. We are aware of no work that specifically explores these mid-range tasks. As well, for barehand control of video timelines, it is unclear whether the user would perceive this task as a direct manipulation task such that a 1:1 or constant mapping would be desirable, or whether some form of pointer acceleration would prove beneficial. Given this lack of related work, we now describe the design of a study to assess acceleration functions for seeking and scrubbing in video timeline control.

2.3 Object Pointing

When we discuss pointing and targeting in human-computer interaction, our assumption is typically that we are discussing Fitts-style positional pointing [25], where an input device – the computer mouse, a finger on a touchscreen, a hand in free space – is mapped onto a discrete location on the screen, the cursor position, and displacement of the input device moves the cursor location via a mapping function. In touchscreen input, this mapping is direct and constant (i.e. a 1:1 control/input device to display movement ratio); with mouse or with more distant arm movements, pointer acceleration is frequently leveraged to support larger range of movement at high speeds and increased precision for targeting at low speeds.

In 2004, Guiard et al. [15] argued that in many interfaces, this 'Fitts-ian' pointing paradigm was inefficient, i.e. that there existed significant non-targetable space on the screen, and that Fittsian style pointing, where you needed to displace the cursor over the entire width and breadth of the screen wasted user effort and required the exchange of a significant amount of data between pointing device and display hardware. This was not precisely a novel observation: many contemporary (with their work) pointing facilitation techniques note the relatively sparse nature of targets on typical displays – McGuffin's *Expanding Targets* work [29]; Grossman's *Bubble Cursor* technique [14]; Asano et al.'s *Delphian Desktop* [2] to name only a few – and leverage this sparsity to either increase the target size [14, 29] or decrease the distance to the target [2], thus improving performance over what would typically be expected of a Fitts's Law baseline for any given targeting task [3].

Guiard et al. [15] formulate object pointing in terms of *information waste*. Essentially, when the cursor leaves a target on the display to move to another target, they note that the system is essentially awaiting one single piece of information: the identity of the new target at which the cursor will arrive. The goal of object pointing as formulated by Guiard is to minimize information waste by ignoring all intervening distance and extraneous pointer information and simply advancing the cursor to the next target once it leaves the confines of the current target/object. To realize efficient selection, Guiard et al. introduce a modified cursor, the ‘timorous cursor’, which they term a ‘void-phobic’ cursor. The timorous cursor jumps from object to object, ignoring the ‘void’ between targets, by analyzing the users desired direction of movement and advancing from target to target along that direction of movement.

Object pointing is significantly different from techniques such as Bubble Cursor or Expanding Targets. In these techniques, users must travel to the voronoi polygon surrounding their desired target, i.e. they still must cover most of the intervening input space. In other words, Bubble Cursor and Expanding Targets increase the effective width of the targets, thus reducing precision requirements. It also differs from the predictive distance hopping of Asano et al. which reduces distance but maintains precision requirements.

With the advent of modern SmartTV displays, the need to support some form of general purpose target selection technique that can be implemented via button-based remote controls has resulted in the adoption of object pointing as a primary input modality for SmartTVs. Users move from object-to-object on the display using directional buttons on a remote control, and this interaction is used to allow access to selectable on-screen objects. Object pointing is used liberally both on single screen displays (e.g. a video player) and on larger, multi-screen, content listings (e.g. a video-on-demand interface).

With the adoption of barehand control, one approach might be to simply eliminate object pointing and return to a traditional Fittsian approach to target acquisition. We would argue that this is both shortsighted and inefficient for three reasons. The first reason is Guiard et al.’s argument on information efficiency [15]. Guiard et al.’s object pointing paradigm was developed on traditional computer displays because of its information efficiency, and their results effectively highlight the benefits of object pointing when target densities are low. Arguably, this information efficiency of input is even more valuable during mid-air interactions because mid-air interaction are much more tiring than mouse or touchscreen input, primarily because the arm is operating in mid-air unsupported [37]. The second is for consistency. It seems obvious to us that traditional remote controls will co-exist for at least some period of time with mid-air input, and preserving a common input paradigm seems valuable to us. Finally, Fittsian style pointing may work well in a certain subset of configurations, i.e. when a single display screen is being navigated and that display screen contains relatively clearly delineated targets and a firm boundary for input. However, there are situations where object pointing simplifies input. As one example, consider tasks such as browsing video on demand interfaces (e.g. Netflix, Prime Video, Disney Plus and others), where only a small portion of a large scrollable region is visible on the display. Fittsian style pointing will require paging, scrolling or clutching to effectively navigate this space [10], all of which mix metaphors and impact input efficiency [18]. As another example, consider 2.5D information, i.e. piles of objects. In dense target configurations such as interfaces that support document piles or other stacked items, rate-based object pointing techniques have proven beneficial in simplifying input [1].

2.4 Mid-air pointing

Since Bolt's work in the 1980s [5], a significant body of work has explored mid-air pointing and input to distant displays. While a full review of all of this past work is beyond the scope of this paper, Koutsabasis et al [23] provided a systematic review on mid-air interaction, and the interested reader is referred to their work. Considering, specifically, pointing, work in this area often focuses on enhancing mid-air pointing precision given the inaccuracy of mid-air pointing. For example, Nancel et al. [32] provided a coarse and precise strategy for 2D pointing at ultra large wall displays. They also point out the limitations of the human visual and motor system in perceiving and acquiring content. Mayer et al. [28] proposed using a model to compensate for the inaccuracy of direct pointing with an offset. Many other systems have leveraged bi-moded input to support targeting (e.g. variants of hybrid pointing [12]) by alternating between direct or constant CD Gain and indirect manipulation.

One challenge with mid-air pointing is acquiring the position of the user's hand with sufficient precision to support precise input. If location detection of the input device is poor, then precise targeting will be compromised due to the need to require larger hand displacements so that movement signals can be discriminated from noisy capture. One approach to this is to add markers to a user, deploy a highly reliable motion capture system such as a Vicon, and to essentially punt on capture challenges [12]. Another approach is to design some active ranging mechanism to detect the users location reliably in mid-air – an approach adopted by the Microsoft Kinect and the Leap Motion. Researchers have explored the use of personal devices [21, 36, 33] to perform mid-air pointing on large displays, including the incorporation of linear or discontinuous CD Gain functions [21]. Furthermore, on-going advances in computer vision algorithms and a lowering of the cost of high resolution cameras has allowed a significant body of recent work to leverage optical tracking algorithms to support barehand input [39].

Based upon the above work, it is obvious that there exists extensive research in mid-air pointing. However, while work on object selection exists, this object selection is – to the best of our knowledge – focused on Fitts-style target acquisition, i.e. point-and-click style input. However, we would argue that understanding object pointing merits exploration with respect to at-a-distance input for two reasons. First, while SmartTV manufacturers including Huawei, Samsung, and LG are introducing mid-air barehand input as a modality to control these displays, it seems likely that these input techniques will, at least initially and maybe even long term, exist in concert with standard remote controls. As such, object pointing may prove intuitive and natural to users because it preserves the object-pointing paradigm that is supported by remote controls. Second, and more generally, the motivation for object pointing was that it reduces the amount of movement – the effort – of the user in controlling an interface. Given the effortful nature of barehand, mid-air input [16], object pointing may be one way to reduce the effort required during target selection tasks that leverage mid-air input gestures.

Chapter 3

Video Timeline Control

3.1 Barehand Input to Smart Displays

As we note in our introduction, manufacturers of Smart TVs are increasingly incorporating barehand, freespace gestures as a mechanism for controlling the display. While several options for gesture capture present themselves, Huawei, Hisense, and Samsung currently use camera-based solution to capture barehand gestures. In this section, we describe the architecture of our gesture capture system. While the gesture capture system is not the focus of this thesis, we include the details to support replicability of our results.

Our gesture capture and recognition system leverages a computer vision application running on HiSilicon's Kirin 990 processors. For experimental set-up, we deployed our test application on a Huawei Mate 30 Pro running Android 9 (API level 28). We used the Mate 30 Pro's front-facing camera to capture movement; by default, the camera captures 1080p video at 30fps.

Our gesture capture system is a four-step process, given an input video frame. Our application first detects the user's face from the image captured by the front-facing camera and generates a virtual gesture activation area just below the user's face. We begin with face detection because face detection is a mature technology, common in modern digital cameras and smartphones. Public domain algorithms are included in open source computer vision toolkits such as OpenCV. We use face position to then identify a gesture activation region. Figure 3.1b shows a user of our system with the user's face outlined in red and the user's hand outlined in blue. A larger blue shaded rectangle in the figure represents the gesture activation area. The size of this gesture activation area is a function of the size of the user's face during face detection (5 face-widths wide by 3 face-widths high).

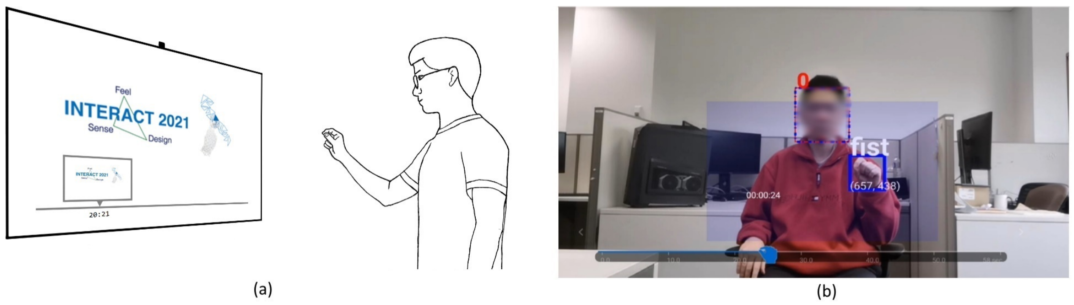


Figure 3.1: Video timeline control for smart TVs. (a) The usage in a real world task. (b) The view of the camera on the smart TV when the user is navigating.

Next, our algorithm specifies a rectangular gesture input area below the face (see Figure 3.1). Hand tracking and gesture input must start within the gesture activation area. By restricting initial tracking to within the activation area, it is possible to significantly reduce false positives in complex backgrounds. It also makes it easier to associate a hand to a face belonging to the same person. For each detected hand, a hand classification and gesture recognition algorithm is run; this gesture algorithm can detect two hand states, pinch open and pinch close, as shown in Figure 3.2. These two gestures allow a simple three-state input model supporting the equivalent of 1) no movement (hand not in frame), 2) mouse move (pinch open with hand moving in activation area) and 3) mouse drag (pinch close with hand moving in activation area) states, analogous to similar states in a computer mouse. From pinch open, a close-open action represents a "click". From pinch close, a pinch open stops a dragging operation. Clutching, as with a computer mouse, is supported during dragging via a pinch-open, move hand, pinch-close action. In Figure 3.3 we show the state diagram of the gestures.

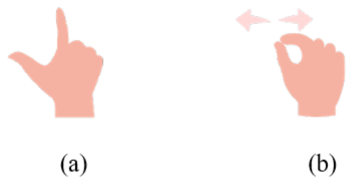


Figure 3.2: The (a) Pinch-open gesture, and (b) pinch-close gestures.

To understand how hand movement is mapped onto screen movement for video timeline control via a CD Gain function, Equation 3.1 shows the relationship between on-screen

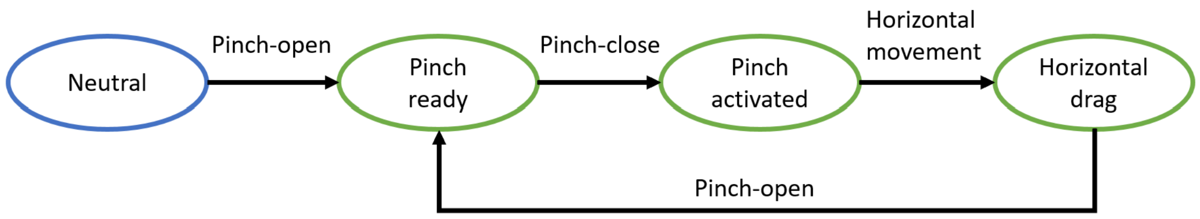


Figure 3.3: State diagram of a complete horizontal dragging.

translation, ΔT, and in-air movement of the hand, Δx.

$$\Delta\mathrm{T} = \Delta x \cdot cdgain \tag{3.1}$$

The CD Gain function can be constant or it can vary with the movement speed of the user's hand.

One challenge with a camera-based solution is to determine movement distance of the hand. Users can sit closer or farther from the display and they may hold their hand at different distances in front of their body while performing input, which results in changes in the perceived size of the movement in the camera view. To create a distance-invariant movement range, we measure movement in units of the user's hand width (HW).

In practice, the above approach works well for supporting barehand timeline manipulation. The algorithm is sufficiently reliable to function across a range of lighting conditions and can easily be deployed in-the-wild for distributed data collection.

3.2 Evaluating CD Gain for Barehand Timeline Control

As we note in the introduction, given the need to support range of motion along a timeline, precision of selection within a timeline, and control of speed to monitor keyframes, it is unclear how best to design a mapping function that maps barehand movement onto timeline manipulation, i.e. an appropriate CD Gain function for timeline manipulation. In this section, we describe an experiment to test three different CD Gain functions (Constant, Linear, and a higher order polynomial function) for three video timeline manipulation tasks.

3.2.1 Participants

Twelve participants aged from 21 to 28 (Mean=24.5, SD=1.55) were recruited (four identified as male) from a local university. Each participant received 50 local currency units for their participation in the experiment.

3.2.2 Apparatus and Interaction

To maintain social distancing guidelines[1], experiments were conducted in participants' homes using disinfected equipment provided by the researchers. The experimental equipment consisted of a Huawei Mate 30 Pro smartphone running our experimental software, a 15 inch portable, freestanding display controlled via USB-c, and a USB-c cable. User hand movement was captured by the front-facing camera on the Mate 30 Pro smartphone, while visual output of the program was displayed on the 15-inch display.

To control the timeline, participants were asked to perform a "pinch open" gesture (Figure 3.2) with either left or right hand in the activation area in front of their chest to enable the timeline. When the program recognised the "Pinch-open" gesture, a scaled slider bar (0 to 7200s, with a scale every 600s) appeared on the bottom of the screen showing the current position of a cursor with a white circle. To manipulate the cursor, the participant brings their index finger and thumb together, i.e. the "pinch-close" gesture (Figure 3.2). When the "pinch-close" gesture was recognised, the cursor changed to a blue rectangle with a tick sound to indicate the cursor was ready to drag. The program gave a tick sound for every 10s-equivalent of movement of the cursor on the 7200s-equivalent timeline. To select a desired time interval, the participant performed the "pinch open" gesture, releasing the blue rectangle within the desired region. Each trial ended either when the "pinch-open" gesture was recognised while the cursor was in the target range or after 10s from trial start. In both cases, a audible confirmation sound was played and the trial ended.

3.2.3 Experiment design

We used a within participant design for the experiment. Participants performed three different tasks for each of the three CD Gain conditions. CD Gain conditions were fully counterbalanced. Tasks were partially counterbalanced across participant (Tasks 1 and 2 were fully counter-balanced; task 3 was always the final task performed).

[1]The experiments were conducted during the Covid-19 pandemic.

CD gain conditions

One challenge in identifying appropriate CD Gain functions is that a large number of functions have been used in past research (see [8] for a review). As well, some past research has used non-continuous functions as CD Gain functions. For example Muller [30] used a minimum CD Gain of 3.0 for mouse speed up to 10cm/s (i.e. constant gain); a maximum gain of 5.0 for mouse speed above 30cm/s (i.e. a second constant gain), and a linear function for mouse speed between 10 and 30cm/s. Our goal was to avoid complex, piecewise, discontinuous functions while probing CD gain functions that are consistent with the research literature and with modern operating systems.

To accomplish this goal, we used a constant gain function as a control condition, and two accelerated gain conditions: a Linear gain function and a non-linear gain function. We hand tuned the form and parameters for our three gain functions through pilot studies involving members of the research team and colleagues (i.e. *person-down-the-hall testing*).

- For our constant CD Gain condition, we chose a CD Gain value equal to 1 timeline width to five HW. In this way, a displacement of five HW of the participant's hand would move the cursor from the beginning to the end of the timeline.
- We implemented our linear CD Gain function as a simple linear function. In the Linear acceleration condition, cdgain was defined as $0.2 \cdot x$ where x is tracked hand speed. Hand movement was calculated in HW, speed in HW/s; on-screen units were calcuated in reference to the timeline width, as with constant CD Gain.
- Finally, for our non-linear acceleration condition, a generalised logistic function (GLF) was used to define the gain (Equation 3.2). In this study, we set A=0, K=7, B=0.2, V=0.05, Q=0.1, C=1, M=5.5. Again, dimensions mimicked constant gain.

$$[h]Y(x) = A + \frac{K - A}{(C + Qe^{-B(x-M)})^{\frac{1}{v}}} \tag{3.2}$$

Our choice of a generalised logistic function was influenced by pointer acceleration curves used in modern desktop operating systems [7] and by recent research in pointer acceleration dynamics [30]. The GLF both replicates very closely the form of commercial CD Gain functions and serves as an elegant, continuous approximation of Muller's [30] piecewise function. We posit that the CD Gain functions we select have significant benefits

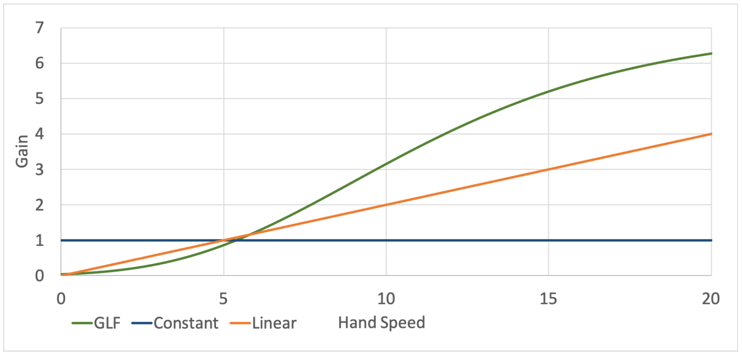


Figure 3.4: CD gain conditions

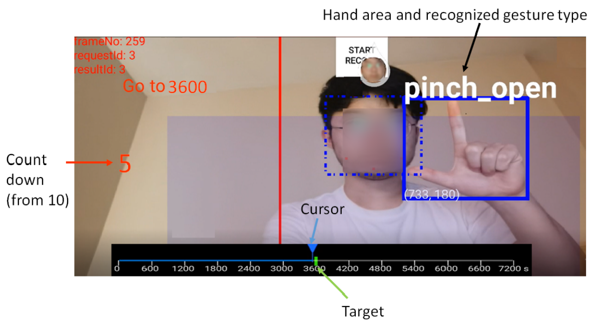
in terms of reproducibility. As Casiez and Roussel note, commercial CD Gain functions are difficult to reproduce exactly [7], whereas ours can easily be replicated using the above information. It is possible that a piecewise function with linear range and max-min discontinuities [30] may serve some performance benefit over our GLF, but, if so, this benefit is not apparent to us as our GLF function can closely approximate Muller's discontinuous, piecewise constant/linear function. Figure 3.4 visualises CD gain by hand speed in each CD gain condition.

Tasks

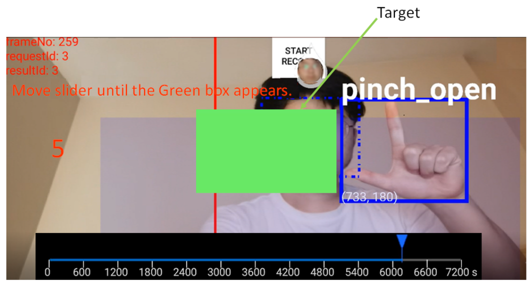
As noted above, there were three tasks: seeking, scrubbing, and a third precise target task. The first two tasks, seeking and scrubbing were counterbalanced; precise time value selection was always the final task. For the first two tasks, seeking (move the timeline cursor to a specific location highlighted on the timeline) and scrubbing (find a hidden along the timeline using simulated keyframes), there were 6 blocks of 9 trials each. The 9 trials consisted of 3 tolerances for the desired timeline location and 3 distances between starting point and desired timeline location. Distances along the timeline were 1200, 2400 and 4800 (out of 7200 maximum units); tolerances were ± 15, 30 and 60 units (out of 7200 along the timeline).

The third task consisted of only one block with 9 trials. The participant attempted to target an exact 1-unit (1 second) point on a timeline with 7200 divisions (analogous to 1 second targeting in a two-hour video).

Task 1: Seeking Our seeking task is designed to simulate a user jumping to a specific known location on the video timeline. In this task, target range was shown in green on the


Hand area and recognized gesture type
frameNo: 259
requestId: 3
resultId: 3
Go to 3600
pinch_open
Count down (from 10)
5
(733, 180)
Cursor
0 600 1200 1800 2400 3000 3600 4200 4800 5400 6000 6600 7200 s
Target

(a) Seeking Task


Target
frameNo: 259
requestId: 3
resultId: 3
Move slider until the Green box appears.
pinch_open
5
(733, 180)
0 600 1200 1800 2400 3000 3600 4200 4800 5400 6000 6600 7200 s

(b) scrubbing Task

Figure 3.5: Experimental conditions.

timeline, and the participants were instructed to move the cursor within the target range as quickly and as accurately as possible (Figure 3.5a).

Task 2: Scrubbing This task assumes a user browses video frames to find a specific scene. In this task, a green square was displayed on the screen (above the timeline) while the cursor was in the particular target range. Unlike the first task, the target range was not indicated on the timeline, so that the the participants had to "scrub" the bar until they found this green target. As with the previous task, there was a tolerance for the desired location. Participants were instructed to stop the cursor at the position where green square was displayed. To help participants locate the target, a yellow square was shown before and after the target range (Figure 3.5b) (at double tolerance). The analog to finding a particular scene in a video is as follows: we often know the context around a desired scene, so we seek proximal key frames (yellow) before homing in on the specific scene (green) we wish to acquire when manipulating a timeline.

Task 3:Precise Timeline Selection Our final task was designed to observe the limits of precision regarding participants' control of the timeline. In this task, participants were instructed to move the cursor to an exact number on the timeline. The number indicating the current cursor position was shown near the cursor above the timeline; the exact number of acquire was given to participants prior to the start of the trial.

3.2.4 Procedure

Participants completed the study individually in their own home. Once participants volunteered for the study, the experimenter fixed a meeting location and provided the participants with a disinfected, reusable bag containing the disinfected smartphone, disinfected monitor, and a disinfected usb-c to hdmi cable. Participants took the equipment to their home and connected with the experimenter via a video call.

After obtaining consent, the experimenter explained the goal of the research, assisted the participant with set-up, and demonstrated the gestures (”pinch-open” ”pinch-close”) and allowed participants to practice move and drag via barehand input using the gestures. This ensured participants were comfortable with providing input to the system and verified that our system was working effectively, important because of the varied environments in which the study was run.

For set-up, participants were asked to place the monitor on a table (preferably a table of standard dining table height) with the smartphone propped against it pointing toward the user. The monitor was to be placed 1 - 1.5m from the edge of the table, with the smartphone propped against the display oriented in landscape mode with the front facing camera pointed forward. Participants then sat in a chair at the edge of the table with their body positioned 0.3m = 0.5m from the table edge. Participants were asked to perform gestures in the space between their face and the display to simplify tracking. Participants were permitted to rest their elbow on the table if they became tired. We also asked participants to preference a plain background behind them to ensure optimal tracking (as a preventative measure to ensure good data collection, though our algorithm seems highly stable regardless of background).

The participants performed the three tasks under a CD gain condition (seeking then scrubbing or vice versa, concluding with the precise selection task). For each task, a trial ended either with successful selection or after a 10 second timeout. If the trial timed out, it was marked as an error and the next trial began.

After finishing all trials under one CD gain condition (117 trials), participants were asked to answer the following three questions on the touch screen of the smartphone using a 7 point Likert scale (1: Strongly disagree, 4: Neutral, 7: Strongly agree); ”I could control the bar precisely”, ”I could control the bar quickly and smoothly”, ”I could control the bar without feeling tired” and ”Overall, I liked this condition”. Then, the participant repeated the same procedure with a different CD gain condition until they completed all three CD gain conditions.

The participant and experimenter remained in a video call until the end of the experiment. The average time to complete all tasks (351 trials) was approximately 1 hour, and the experimenter encouraged the participants to take breaks between blocks to avoid fatigue. At the end of the session, the experimenter collected general feedback and then set up a meeting to collect the experimental equipment from the participants in a physically distanced manner.

3.2.5 Data Collection

Considering all factors above, each participant completed 351 trials: ((3 distances x 3 tolerances x 6 blocks x 2 tasks x 3 CD gain conditions) + (9 trials x 1 task x 3 CD gain conditions)) in total. For 12 participants, we collected 4212 data points for analysis.

Our program recorded time for successful positioning and an error when positioning time exceeded 10 seconds. Likert question data was stored on the smartphone and downloaded after each participant.

3.3 Results

The primary goal of our study is comparative, i.e. to understand whether a constant versus non-constant CD Gain function can enhance barehand control during seeking and scrubbing video timeline tasks. Because this goal is formative, not summative, we have no hypotheses about which CD Gain function is best. Therefore, we present a statistical analysis of our results comparing our control condition, a constant gain, to linear and non-linear (generalized logistic regression) CD Gain functions organized by task.

3.3.1 Seeking

Accuracy

Figure 3.6 (left) shows the error rate (the proportion of trials participants could not acquire the target within 10 seconds) of the seeking task. The result of repeated measures ANOVA showed significant main effects of CD gain ($F_{2,22}$=4.49, p<.05), Target distance ($F_{2,22}$=4.28, p<.05), and target width ($F_{2,22}$=6.89, p<.001), and a significant interaction between target distance and width ($F_{4,44}$=2.70. p<.05). The result of post-hoc tests (using Holm's correction) showed that error rate in the Constant CD gain condition was higher than that in the Linear condition (t_{11}=2.81, p=.051, borderline but with significant RM-Anova for CD Gain).

Speed

Figure 3.6 (right) shows the median task completion time (TCT) of the seeking task for successful trials. The result of repeated measures ANOVA showed significant main effects of CD gain ($F_{2,22}$=9.27, p=.001), Target distance ($F_{2,22}$=36.11, p<.001), and target width ($F_{2,22}$=78.96, p<.001), and a significant interaction between CD gain condition and target distance ($F_{4,44}$=2.72. p<.05). The result of post-hoc test (using Holm's correction) showed that median TCT in the GLF CD gain condition was significantly longer than that in the Linear condition (t_{11}=4.11, p<.01) and Constant condition (t_{11}=-3.24, p<.05).

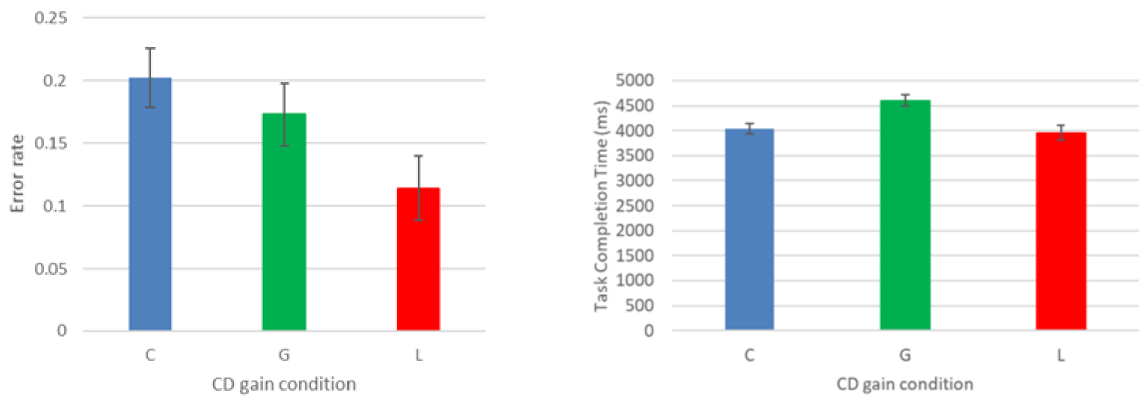


Figure 3.6: Mean error rate and Task completion time by CD gain in Seeking Task. Error bars represent the std. C: Constant, G:GLF, L:Linear.

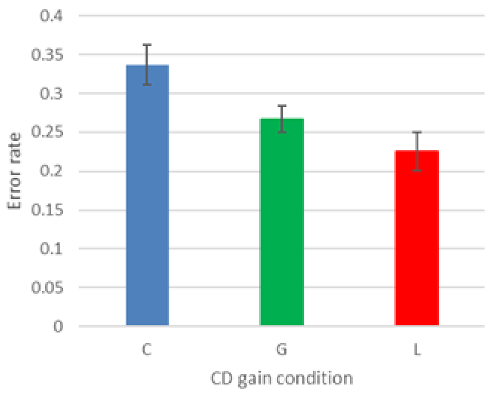
3.3.2 Scrubbing

Accuracy

Figure 3.7 (left) shows the error rate of scrubbing task. The result of repeated measures ANOVA showed significant main effects of CD gain ($F_{2,22}$=4.99, p<.05), Target distance (F=15.52, p<.001), and target width ($F_{2,22}$=33.73, p<.001), and a significant interaction between target distance and width ($F_{4,44}$=6.02 p<.001). The result of post-hoc test (using Holm's correction) showed that error rate in the Constant CD gain condition was higher than that in the GLF CD gain condition (t_{11}=2.97, p<.05) and in the Linear condition (t_{11}=2.44, p=.066, borderline).

Speed

Figure 3.7 (right) shows the median task completion time (TCT) of the scrubbing task. The result of repeated measures ANOVA showed significant main effects of Target distance ($F_{2,22}$=44.43, p<.001) and target width (F=100.31, p<.001), but not CD gain. We also observed significant interactions between CD gain condition and target width ($F_{4,44}$=3.16. p<.05), between target distance and width (3.65, p<.05), and a three-way interaction among CD gain, Target distance and target width ($F_{8,88}$=4.02, p<.001). The result of post-hoc test (using Holm's correction) showed that median TCT in the GLF CD gain condition was significantly longer than that in the Linear condition (t_{11}=4.11, p<.01) and in the Constant condition (t_{11}=-3.24, p<.05).



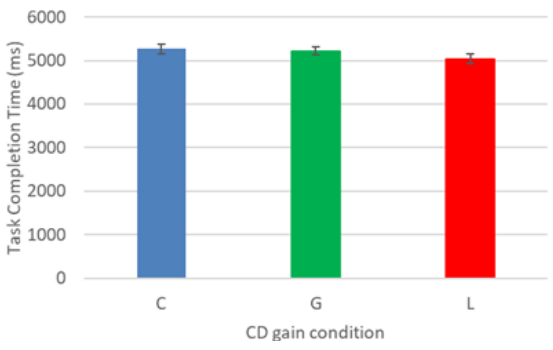


Figure 3.7: Mean error rate and Task completion time by CD gain in Scrubbing Task.

Subjective ratings

Figure 3.8 shows the mean score of subjective ratings by CD gain condition in Task 1 and Task 2. The result of repeated measures ANOVA showed no significant main effect of CD gain condition on all measures.

3.3.3 Precise Time Selection

Recall that the goal of our third task was to select, as accurately as possible and within a ten-second timeout, an exact value on a timeline with 7200 divisions (i.e. to attempt to select a specific one second location in a two hour timeline within 10 seconds). We note

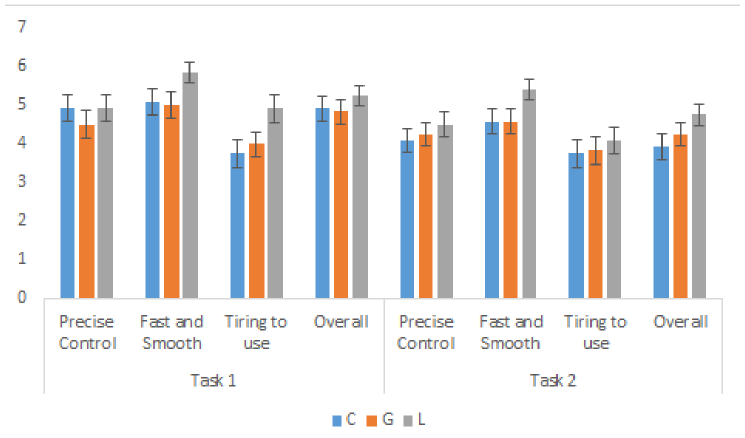


Figure 3.8: Mean subjective ratings on CD gain conditions with standard errors. Higher score represents a better performance.

that, because this involves sub-pixel selection on a 4K resolution display, it is an exceedingly difficult task. Selecting a 1-second interval on a two-hour movie is challenging with a computer mouse on a computer display, so we expect a high error rate. To assess performance on this task, we use two measures: accuracy (i.e. how frequently did participants select the exact second) and average gap (i.e. how far, on average, were participants from the 1s interval when the task completed).

Accuracy

Mean accuracy in Task 3 was 0.03 (SD=0.05), i.e. 97% error rate, in the Constant CD gain condition, 0.31 (SD=0.21) in the GLF CD gain, and 0.31 (SD=0.16) in the Linear CD gain condition. In other words, participants had only a 3% success rate selecting a 1s interval with Constant CD gain, whereas their accuracy was more than 10 times higher (31%) with pointer acceleration.

Average Gap to Target

Selecting an individual second in a two-hour video – with an attendant 31% success rate – may not be necessary. Perhaps selection within two or even three seconds is acceptable – especially given that selecting a 1-second interval using a mouse along a timeline is also exceedingly difficult. A more important question is *how closely* a 1-second interval can be targeted using barehand, in-air input with an appropriately-chosen CD Gain.

To analyze this, we examined the distance – the gap – between the target one second interval and the actual position selected when a trial finished as a measure of precision in task 3. For those trials which were successful, the gap was 0. To calculate this value, we first calculated median gap in the nine trials of each participant under the same target condition (distance and width), then calculated the mean of all 12 participants. The mean gap in the Constant CD gain condition was 5.92 (SD=3.01), in the GLF CD gain condition it was 1.33 (SD= 1.11) and in Linear CD gain condition was 1.50 (SD=1.50). The result of repeated measures ANOVA showed a significant main effect of the CD gain condition ($F_{2,22}$=28.83, p<.001), where the gap to the target in the Constant CD gain condition was significantly larger than that in Linear condition (t_{11}=4.78, p<001) and GLF condition (t_{11}=6.44, p<.001).

3.4 Discussion

In our introduction, we posed two primary questions: 1) What CD Gain function is most effective in barehand control? and 2) How well do each of these different variants of CD Gain functions work for both seeking and scrubbing tasks? Considering the first question, it is reasonable to ask whether mapping of hand movement onto on-screen movement is more analogous to direct manipulation – with its attendant 1-to-1 mapping or constant gain – or mouse manipulation – where pointer acceleration, i.e. a non-constant gain function, can provide targeting benefits [8]. Furthermore, in the absence of data it is also unclear whether practical gain functions exist to support pointing tasks given potential requirements for precision (e.g. sub-pixel accuracy on a 4K display) and range [31]; in the absence of data it is also unclear whether seeking and scrubbing tasks can be optimally supported by a similar CD Gain function.

Synthesising our results, and considering the performance of various CD Gain functions, we note the following. First, linear CD Gain exhibits lower error rate than constant gain for both seeking and scrubbing during timeline access. Second, a Linear Gain function was faster than a GLF gain for seeking tasks. Finally, our results argue that both Linear and GLF Gain functions allowed higher-precision targeting of a 1-second-sized interval than constant gain. Median average precision for constant gain was 1/10th the precision of Linear or GLF gain. Together, these results indicate an advantage of non-constant CD Gain for both seeking and scrubbing tasks. For example, for both seeking and scrubbing, Linear Gain has better error rate than Constant gain and lower task completion time (higher speed) than GLF gain.

While the overall goal of our studies was timeline manipulation, we note that timeline manipulation is only one of a number of potential targeting and input tasks in smart display/Smart TV input. Video playback requires support for a number of commands (play/pause, volume control) alongside timeline manipulation. The results of our experiments can apply to, for example, volume manipulation, which has similar characteristics to timeline manipulation; however, volume ranges are typically less granular than are timeline manipulations (50 or 100 levels vs thousands of seconds).

Our results may also be applicable to targeting in general on Smart TVs. However, on-screen widget targeting is much more permissive than timeline manipulation. Consider, for example, a Smart TV display with a number of on-screen widgets that a user would like to target. While the amplitude of movements on the display (the distance of movements) are of the same scale as timeline manipulations, it is not typical that on-screen widgets require pixel level (or even sub-pixel level) targeting, i.e. precision requirements are more relaxed because widgets typically span multiple pixels. As well, widget targeting can often benefit from pointing facilitation techniques [3, 14, 29] that can leverage inter-widget whitespace to increase the effective size of widgets or the cursor, but timeline controls have no inter-target whitespace because every element along the timeline, up to even the sub-pixel level in our experiment, is an allowable target. We would argue that timeline manipulation is among the more challenging targeting tasks to support. We would argue, however, that the ability of barehand manipulations to support near pixel-level targeting (median error rate for precise selection/task 3 is at the pixel-level for both Linear and GLF gain functions) provides evidence that barehand point-and-click style targeting is one effective option for smart TV control across a variety of tasks, even those requiring very high precision.

The caveat to point-and-click style targeting for Smart TVs is that it is unclear if smart TVs should use point-and-click style targeting in all interactions. As one example, video-on-demand streaming services use directional navigation plus object pointing to support elegant scrolling, and it may be that this represents a more effective, controllable interaction for access to multiple pages (both vertical and horizontal) of content. Menuing systems on smart TVs may benefit from gesture-style input (e.g. marking menus) rather than multiple layers of point-and-click widgets. Even non-timeline video playback may benefit from more simplified gesture input rather than requiring the user to target different on-screen widgets.

However, despite these factors, video timeline control is a common task. As we note earlier, because of the complexity of this task with respect to precision, and because of the confound between seeking and scrubbing, researchers continue to seek enhanced ways to balance range and precision. Our work provides important support to this domain by highlighting the potential of effectively designed CD Gain to support barehand gestural dragging as a mechanism to effectively manipulate video timelines.

3.4.1 Limitations and Study Design Factors

As with any study, there exist limitations to this study.

Some limitations are debatable. As one example, one can pose the question of how representative our experimental tasks are of real-world tasks. Our initial seeking task asks participants to acquire a specific range which is highlighted on the timeline; our scrubbing task uses only a green rectangle on the display rather than a full scene; and our 1-second-level precise targeting task on a 7200 second-equivalent timeline is perhaps more precise than a user would require. We made these choices consciously. We varied the tolerance of our first seeking task because we recognise that users vary in their needed precision. We used a green square to indicate desired scene when scrubbing to eliminate visual perception confounds associated with scene recognition. Finally, we use a second-level precision attempt to truly test the limits of different CD Gain functions. Our goal with each of these decisions was to balance experimental and ecological validity, but these choices are debatable.

The experiment design, itself, also has limitations. As a simple example, our display, provided to participants, is both smaller and closer than a typical television. Our goal in providing equipment was to increase control and to limit risk to participants of installing software on their own personal devices (ethics review at our institution is reluctant to allow installation of software on participants' personal devices due to privacy and/or security risks). This drove the need to provide our own hardware to participants in a responsible manner given the context of a global pandemic. However, this fixed hardware represents a risk to generalizability. Specifically, we specified distances for our study to accommodate for the smaller (than typical smart tv) portable display size with closer distance. A 15-inch display at 1 to 1.5m distance, 0.25 radians of visual arc, is similar in visual arc to a 32 inch at 2 to 3m or a 65-inch at 4 to 6m. However, small and or close displays may alter aspects of user behaviour in unanticipated ways, even if the displays appear visually similar in scale.

Another potential limitation is latency. Via an optical camera capturing both user movement and the 15" display, end-to-end latency in our experimental set-up was measured at 136ms. Approximately 2/3 of this latency (90ms) is from image processing and movement mapping in Android, 20ms is due to USB-C communication, monitor response

(4ms) and refresh (60Hz), and the remainder is due to camera capture latency from the Mate30 Pro front camera (30fps). Latency could represent a risk to validity, especially if camera-equipped commercial smart tvs have lower or higher latency.

Finally, our participant population is a limitation. Our participants were younger and were drawn from a highly educated, relatively affluent demographic. This was not by design, but may have been a result of recruitment challenges and trepidation of older/less healthy individuals, both of which can be attributed to the Covid-19 pandemic concurrent with this study. Older participants may struggle more significantly with issues of precision and reach, which might impact time and/or errors.

3.5 Summary

Motivated by the introduction of barehand control for Smart TV displays, this paper examines the effectiveness of pointer acceleration in barehand, distant, video timeline control. We evaluate three CD gain variants: a constant function (no pointer acceleration, analogous to direct manipulation) and two different gain functions implementing pointer acceleration, a continuous linear function and a generalised logistic regression function. We find overall benefits to non-constant CD Gain in speed, error, and precision, particularly for our linear CD Gain function.

Alongside to video timeline control, another challenging task is targeting, which is going to be explored in the next chapter.

Chapter 4

Mid-air Pointing

In the previous chapter, we studied timeline control, a form of high precision input. However, users of smart tvs also need to perform targeting tasks to acquire discrete targets on displays, including objects in video on demand interfaces, widgets in video playback, and apps that are available on smart tvs. The primary paradigm for this is object pointing, which motivates our research into how best to support object pointing on smart tvs. In this chapter, we design five different mappings ,compare three of them and provided design guide base on the result.

4.1 Object-Based Pointing Via Barehand Input

In our analysis of object pointing in SmartTVs, we leverage Guiard et al.'s [15] formulation of object pointing in terms of the manipulation of a timorous cursor. We identify three different potential paradigms for timorous cursor repositioning: discrete, positional, and rate-based. Discrete movement involves performing a discrete atomic action to advance the timorous cursor to the next target; an example of this discrete action is the directional buttons on remote controls. Directional buttons on remote controls are a highly information efficient mechanism for supporting repositioning: A single action, the press of a button, advances the cursor, yielding an optimal information transfer between controller and display. Positional repositioning was used in Guiard et al.'s original formulation of object pointing. Leveraging a positional input device, in their case the computer mouse, the direction of mouse movement was analyzed to step to the subsequent object along the direction of movement. When the mouse stopped moving, the timorous cursor's movement also stopped. Finally, rate-based control involves joystick-style movement: the user indicates a scrolling direction with their input device and the timorous cursor continues to move in that direction until the input device returns to a neutral position.

All of the above techniques, discrete, positional, and rate, are commonly found in interfaces to smart displays depending on the affordances of the input device. Typically, the affordances of the input device dictate mapping (e.g. traditional button-based remote controls and joysticks have natural mappings onto discrete and rate-based input respectively). However, it is possible to map barehand mid-air input onto any of discrete, positional, or rate-based mapping. For example, hand position could be mapped onto virtual buttons, and the user could select those buttons repeatedly to step and move; changes in hand position could be mapped to movement of a timorous cursor, i.e. a positional mapping; or one could set a neutral hand position and then implement rate-based directional control depending on offset from neutral.

In this section, we outline our gesture capture system to provide context for our mappings. Next, we describe our implementation of these three mappings.

4.1.1 Mid-Air Input Capture System

Our mid-air capturesystem leverages a computer vision application running on HiSilicon's Kirin 990 processors. For experimental set-up, we deployed our capture system on a Huawei Mate 30 Pro running Android 9 (API level 28). We used the Mate 30 Pro's front-facing camera to capture the movement; by default, the camera captures 1080p video at 30fps. During our experimental set-up, described in the next section, we combined the Mate 30 Pro with a large display via usb-c to support experimental validation.

Our gesture capture system can be broken into five steps, given an input video frame. A screencapture of its functionality is shown in Figure 4.1. First, our system detects the user's face from the image captured by the camera. Public domain algorithms for face detection are included in open source computer vision toolkits such as OpenCV, making this step robust and low-cost. Second, we use face position to identify a rectangular gesture activation region. This activation area is positioned below the user's chin and centrally aligned with the user's body. To control for distance between camera and user, the size of this gesture activation area is a function of the size of the user's face during face detection (5 face-widths wide by 3 face-widths high). Third, we capture the user's hand within the gesture activation area. Hand tracking and gesture input must start within the gesture activation area, which makes it is possible to significantly reduce false positives in complex backgrounds. It also helps associate a hand to a face belonging to the same person. Fourth, for each detected hand, a hand classification and gesture recognition algorithm is run; this gesture algorithm can detect two hand gestures: pinch open and pinch close, as shown in Figure 3.2. Fifth and finally, these two gestures are used to generate a simple three-state input model supporting the equivalent of 1) no movement (hand not in frame), 2) hand tracking (pinch open) and 3) drag state (pinch close with hand moving), analogous to similar states in a computer mouse. From pinch open, a close-open action represents a "click". From pinch close, a pinch open stops a dragging operation. Clutching, as with a computer mouse, is supported during dragging via a pinch-open, move hand, pinch-close action. In Figure 4.2 we show the state diagram of the gestures.

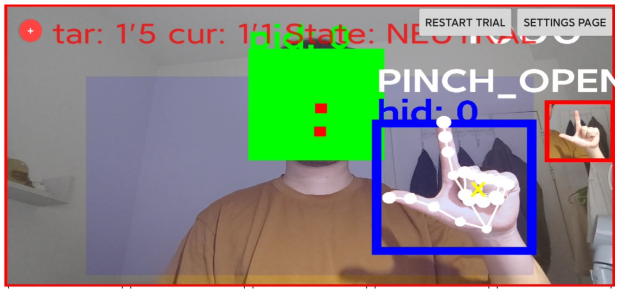


Figure 4.1: The input capture system for mid-air input.

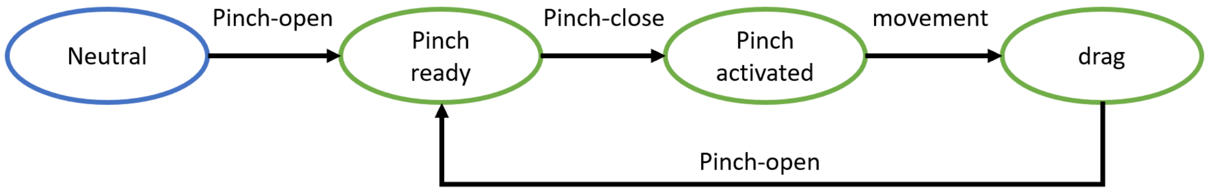


Figure 4.2: State diagram of a complete horizontal dragging.

One challenge with a camera-based solution is to determine movement distance of the hand. Users can sit closer or farther from the display while performing input, which results in changes in the perceived size of the movement in the camera view. To create a distance-invariant movement range, we measure hand movement in units of the user's head width (HW).

In practice, the above approach works well for supporting barehand target manipulation. The algorithm is sufficiently reliable to function across a range of lighting conditions and can easily be deployed in-the-wild for distributed data collection, a particularly important concern given ethical prohibition by our research institution on in-person research due to the Covid-19 pandemic.

4.1.2 Mappings

Discrete Mapping

Discrete mapping leverages an atomic action to perform object pointing. In Figure 4.3, we show two variants of discrete mapping: one that uses only atomic steps and a second that includes paging. To control this mapping, the user raises their hand in front of their body, which activates neutral position tracking, the blue dot. As the user moves their hand up, down, left, or right, the corresponding button is highlighted. By performing a pinch close, pinch open operation, the selected target is advanced in the appropriate direction, exactly similar to a traditional remote control. When the user attempts to step outside the confines of the page, the page is scrolled in the appropriate direction horizontally or vertically. In the paging example, Figure 4.3 at right, a more distant movement highlights the second mark, which pages in the indicated direction, i.e. repositions the view by five targets horizontally or three targets vertically.

Movement mapping is configured so that a displacement of 0.75 headwidths aquires the first "button" to step. For the paging example, an additional 0.5 headwidths results in acquiring the corresponding paging button.

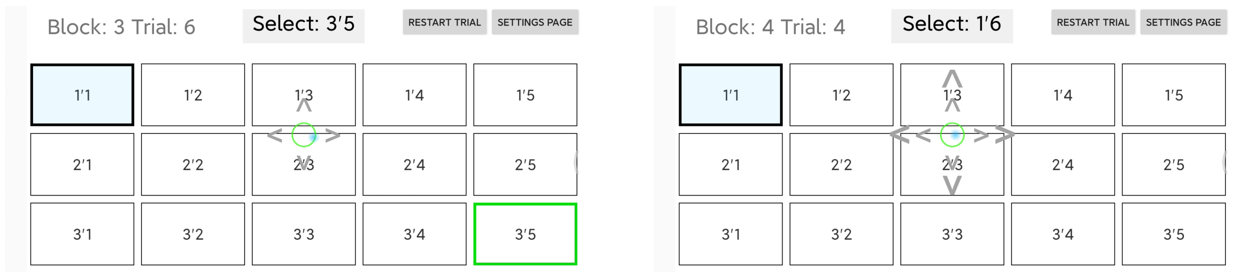


Figure 4.3: Discrete Mapping Layout. The circle in the middle represents select button. Four arrows around the circle are stepping buttons. The four double arrows around the four arrows in the right figure represent paging buttons, which could move the focus by one page.

Positional Based

Figure 4.4 shows the interface for both the position-based and rate-based object pointing mappings. To implement position-based mapping, on pinch close once the hand moves further than a fixed threshold (in our case it is 1/6 head width) in one direction, the focus will move to the next target in the direction of hand movement. To make users aware of the current progress and direction, we show a blue dot inside the target, representing the location of Guiard et al.'s 'timorous cursor' [15].

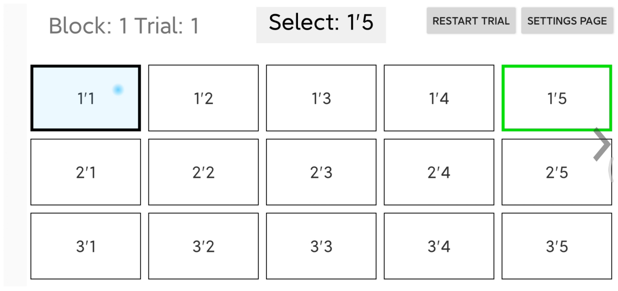


Figure 4.4: Positional Based and Rate Based Layout. The Current focus is highlighted with a thick black border. The blue dot inside it represents the direction and

Rate Based

Similar to a joystick on a game controller, rate based mapping allows users to control the progress by mapping displacement of the cursor from a neutral position onto a scrolling speed. From neutral, as with position-based mapping, users perform a pinch close gesture, then move the hand to control the speed of the target scrolling. Distance from neutral (and hence speed) is represented by a blue dot. The speed is based on the relative position of the start moving position and current position, calculated as shown in Equation 4.1.

$$Rate(d) = \begin{cases} 0 & d < 0.15 \times Head\ Width \\ 0.3d & d \geq 0.15 \times Head\ Width \end{cases} \tag{4.1}$$

4.2 Pilot Evaluation

We performed a series of informal pilot studies within our organizations. Participants performed a series of targeting tasks, selecting targets on both the current screen and by scrolling both horizontally and vertically. During these pilot studies, we experimented with parameters (for example the 0.15 head width measurement was tuned based upon a pilot study). We also collected participant feedback of techniques and performed a series of timed evaluations of selection for a small number of targets.

Alongside tuning values for our parameters, we identified two additional take-aways. First, for discrete mappings, the increased fatigue of holding one's hand in the air and performing repeated discrete or paging operations was fatiguing and felt awkward for participants. It was also very slow.

Second, our pilot study participants noted that the positional mapping worked well on a single screen but not when they needed to scroll. Scrolling resulted in longer reaching, which was tiring, and it also resulted in a need to clutch.

4.3 Evaluating Mid-Air Object Pointing Mappings

To evaluation mappings for object pointing, we conducted a full factorial evaluation of three different mid-air object pointing mappings for target selection on smart displays: a positional mapping, a rate-based mapping, and a hybrid mapping.

4.3.1 Evaluated Mappings

As we note in our pilot study, both positional and rate-based mappings of object pointing exhibit advantages. In particular, while positional mapping seems to provide advantages for input when all targets are presented on a single display, when scrolling is required rate-based mapping may be more effective.

While the above observation motivated our use of positional mapping and rate-based mapping, we also created a hybrid position+rate mapping, again based on feedback from our pilot study. Position+rate mapping is similar in behaviour to a standard computer mouse in a scrollable interface: within the current screen, it is positional; however, when scrolling is required it leverages rate-based mapping.

Position+rate mapping works as follows. Within the current display screen, 1/6 head-width movement in any vertical or horizontal direction results in a target shift. If the user attempts to move beyond a target on the edge of the screen, then the interaction switches to a rate-based mechanism where increasing distance from the display increases speed. Reversing direction immediately stops scrolling, a feature we implemented as a result of pilot testing where participants required an ability to halt rate-based scrolling quickly to avoid overshoot. Stopping scrolling reverts immediately to position-based mapping for the current screen, and rate-based mapping can be reactivated by moving off the edge of the screen, i.e. moving to the next target off the current screen.

Overall, therefore, our independent variable is mapping, and it has three levels: positional; rate; and a hybrid position+rate mapping.

4.3.2 Layout

The experiment layout is designed as in Figure 4.5. We adopt this layout from online video service providers such as Netflix, Hulu and Amazon Prime Video. All rows are independent from each other, which means horizontal scrolling of one row is independent of others. There are 10 rows × 30 columns in total in the interface and 3 rows × 5 columns are shown on the display. The top centre text indicates the target for the current trial. The target is also highlighted with a thick green border. The current focus is highlighted with a thick black border. The Restart Trial button is used to restart the current trial in case of error in input. Finally, the Setting Page button allows a participant to quickly go to a specific point in the trial, for example if they were unable to complete the experiment and wanted to re-engage with the experiment after stepping away.

4.3.3 Participants

Twelve participants aged from 21 to 31 (Mean=24.5, SD=2.47) were recruited (Three identified as female) from a local university. Each participant received $30 CAD for their participation in the experiment.

4.3.4 Apparatus

To maintain social distancing guidelines, experiments were conducted remotely using disinfected equipment provided by the researchers. A Mate 30 Pro smartphone running our

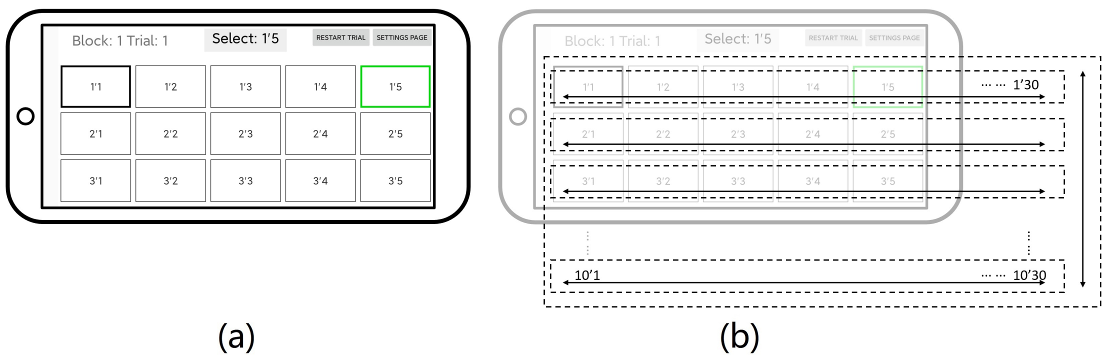


Figure 4.5: The UI of experiment platform. (a) User's view. The target is indicated in the top centre text, and also highlighted with a thick green border. The Current focus is highlighted with a thick black border. (b) In the UI, there are 10 rows × 30 columns in total and 3 rows × 5 columns are shown in the display. Each row can be scrolled left and right independently.

experimental application was used in the experiment. User hand movement was captured by the front-facing camera on the Mate 30 pro smartphone.

4.3.5 Experiment Procedure

We used a within participant design for the experiment, and mappings were fully counter-balanced. Equipment was provided to participants. Participants took the equipment to their own home or other space where the experiment could be conducted (e.g. a dormitory room). Participants then called the experimenter, set-up assistance was provided, and a test capture was performed.

Once experimental equipment was configured in the experimental space being used by participants, participants were asked to watch an introductory slide deck on the experiment and mappings. Then they could start the experimental application, and the application was pre-configured to provide conditions in the correct order.

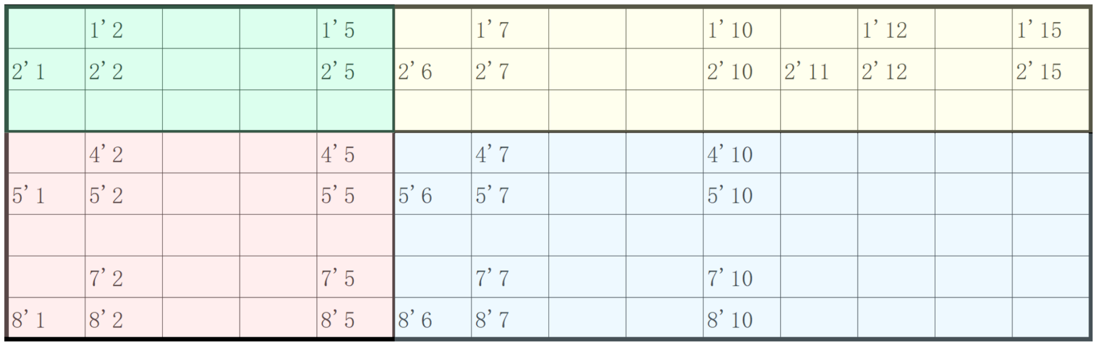
For each mapping, participants completed one practice block with 10 trials and one experimental block with 40 trials. This yields 3 mappings × 1 block × 40 trials = 120 data points per participant, or 1440 trials for 12 participants in total.

To balance different levels of scrolling, we divided the layout into four areas: top-left, top-right, bottom-left and bottom right, shown in Figure 4.6 and defined as follows:

- The top-left region is the initial set of 5 $\times$ 3 targets that are displayed without scrolling when the experimental trial started. If the goal target was one of these targets, then no scrolling was needed in the trial.
- The top-right and bottom-left regions are the top three rows of targets beginning with column 6 and the leftmost five columns beginning from row 4, respectively. These regions represent regions that can be accessed via only horizontal or only vertical scrolling.
- The bottom-right region is the remaining ‘quadrant’ of targets. Accessing targets in this region required both vertical and horizontal scrolling.

To balance the difficulty of our experimental trials, each area contains 10 targets. The actual target items used in the experiment are shown in 4.6. Each target in the top-left area was chosen twice in each block. All other areas’ targets (there were 10 in each area except for top-left) were selected once.

While target identity was fixed, target order was semi-random within each block.



	1’ 2			1’ 5		1’ 7			1’ 10		1’ 12			1’ 15
2’ 1	2’ 2			2’ 5	2’ 6	2’ 7			2’ 10	2’ 11	2’ 12			2’ 15
	4’ 2			4’ 5		4’ 7			4’ 10					
5’ 1	5’ 2			5’ 5	5’ 6	5’ 7			5’ 10					
	7’ 2			7’ 5		7’ 7			7’ 10					
8’ 1	8’ 2			8’ 5	8’ 6	8’ 7			8’ 10					

Figure 4.6: Target chosen in the experiment. Each item in the top-left area was chosen twice in each block. The others were selected once.The top left quadrant is fully on-screen at the beginning of each trial.

The target text and thick green border would appear one second after the trial started. Participants were asked to put their hands down at the beginning of each trial until the target was prompted. To select the target, participants were asked to perform a pinch-open gesture to activate the system. When the system recognized the pinch-open gesture, the background of the current focus switched to blue. Participants could then move the focus based on different mappings. To confirm the selection, participants performed a combination of pinch-open, pinch-close, pinch-open gesture to select the target, which is similar to the click gesture. Participants completed a NASA-TLX questionnaire after each mapping was completed, and participated in a semi-structured interview at the end of the experiment.

4.4 Results

In the following analysis, we leverage repeated measures ANOVA (RM-ANOVA) to test for main effects. A Shapiro-Wilk test was applied to data and, if necessary, normality was corrected via a Box-Cox transform. If sphericity assumptions were violated, Greenhouse-Geisser corrections were applied. Holm's correction was used to control for multiple comparisons in post-hoc tests.

4.4.1 Task completion time

Figure 4.7 shows the task completion time both overall and broken out by each of the four quadrants. To minimize the instability of recognition, the task completion time was calculated from when the participant first starts moving their hand to the time the focus stops moving before the click gesture. The result of repeated measures ANOVA showed no significant main effects of mapping ($F_{2,22}$=0.28, p=.75). However, there was an interaction between mapping and area. For different areas (top-left, top-right, bottom-left, bottom-right), we observed a significant main effects of mapping in the top_left quadrant ($F_{2,22}$=8.64, $p<.001$) and top_right quadrant($F_{2,22}$=3.60, $p<.05$) . The result of post-hoc tests (using Holm's correction) showed that Positional Mapping is significantly faster than Rate ($p<.01$) and Positional + Rate ($p<.0001$) mappings in top_left quadrant and Positional Based is significantly faster than Rate Based in top_right quadrant ($p<.05$).

4.4.2 Target Crossings

Alongside task completion time, we also analyzed the number of times the participants crossed the target and had to return. Zero crossings would indicate that the participant had acquired the target without overshooting and returning. We looked at both horizontal and vertical crossings.

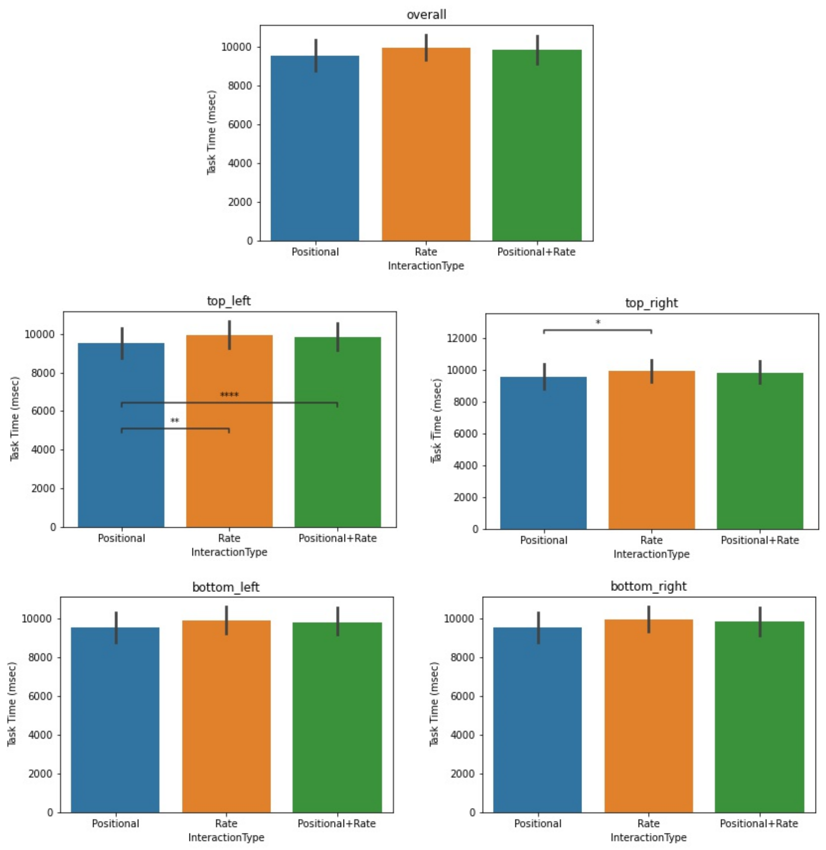


Figure 4.7: Mean task completion time.

Horizontal Crossing

Figure 4.8 shows the number of horizontal crossings both overall and for each region. The result of repeated measures ANOVA showed significant main effects of mapping ($F_{2,22}$=61.19, p<.0001). For different quadrants, the result of repeated measures ANOVA showed significant main effects of mapping in the top_left quadrant ($F_{2,22}$=10.15, p<.0001), bottom_left quadrant ($F_{2,22}$=5.62, p<.01), top_right quadrant ($F_{2,22}$=38.41, p<.0001) and bottom_right quadrant ($F_{2,22}$=26.62, p<.0001). The result of post-hoc tests (using Holm's correction) overall and an area-based analysis also shows that Rate mapping outperforms other mappings. Positional+Rate, the hybrid mapping, performed worse overall and was significantly outperformed by one or both other mappings in each region.

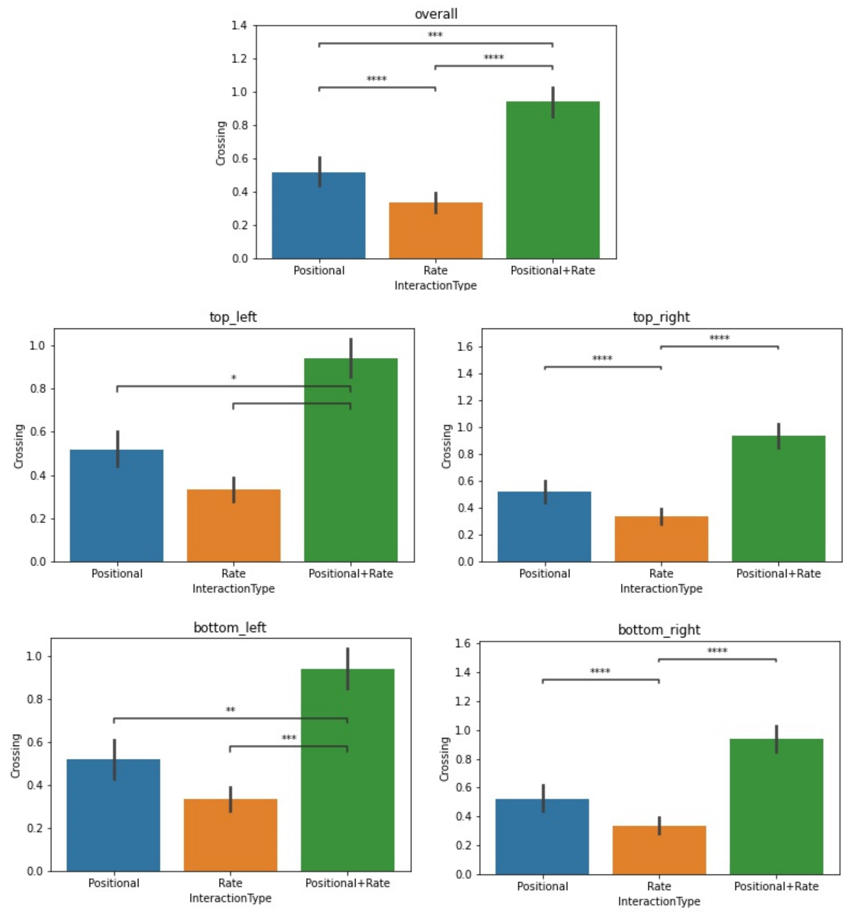


Figure 4.8: Horizontal crossings.

Vertical Crossing

Figure 4.8 shows the number of vertical crossing of the overall and four quadrants. The result of repeated measures ANOVA showed significant main effects of mapping ($F_{2,22}$=4.22, $p<.05$). The result of post-hoc tests (using Holm's correction) showed that Positional + Rate Mapping has more vertical crossing than Rate based ($p<.05$). Rate mapping and Positional mapping do not differ by a statistically significant margin (see Figure 4.9).

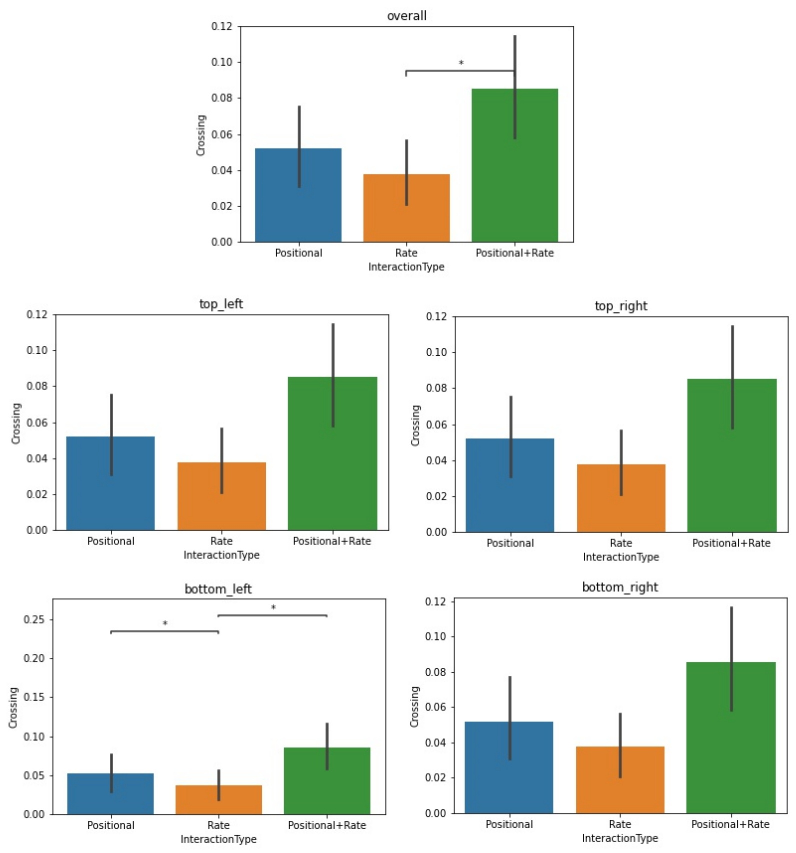


Figure 4.9: Vertical crossings.

4.4.3 Clutching and NASA-TLX Data

Both an analysis of clutching and the NASA-TLX data were not significant. For completeness, we include these data in Figures 4.10 and 4.11.

4.4.4 Interview

In our semi-structured interview, we explored participants preferences for the different mappings and the strengths and weaknesses of the different mappings. Interestingly, despite it marginally better performance in time, participants were quite negative on position mapping. Participants (P1, P4, P5, P7, P10) noted the it required too many clutches (despite the lack of significance in our quantitative data) and that the long reach felt awkward.

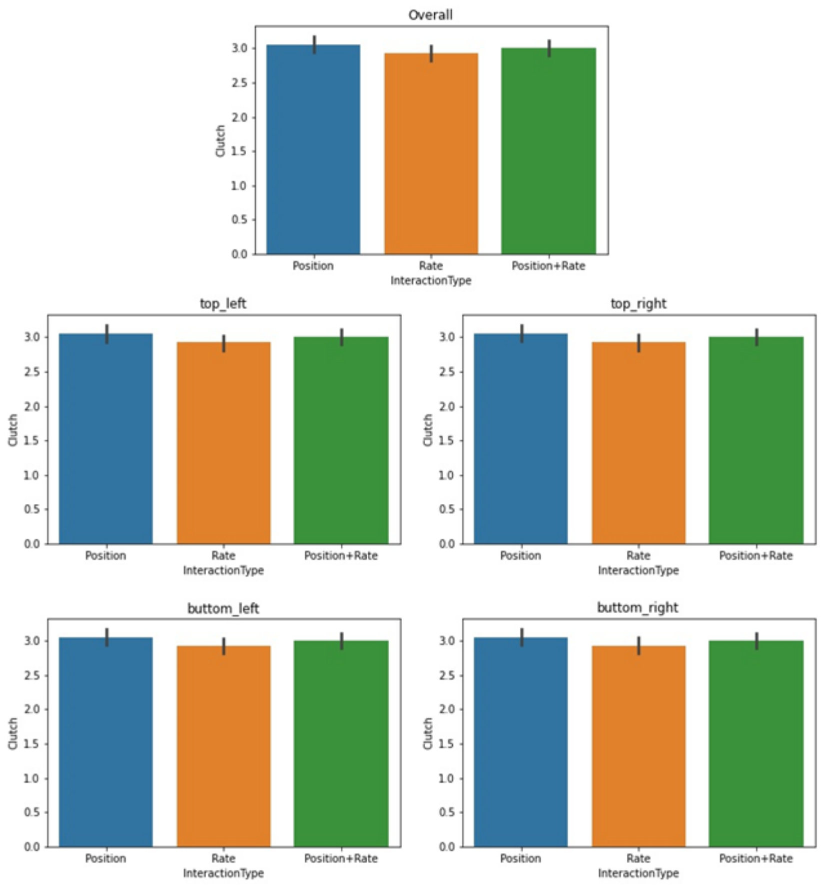


Figure 4.10: Mean number of clutches during input and per quadrant. Not significant.

In contrast, Rate based mapping was reviewed favourably by our participants, perhaps due to effective tuning of scroll speed. Participants (P1, P3, P5, P6, P8, P9, P10, P11, P12) indicated that the Rate Based strategy was comfortable to use and easy to predict. They did note a learning curve for rate-based mapping, highlighting the fact that it was hard at the beginning and requiree some practice (P1, P3, P6, P9, P10, P12). Switching the direction during overshoot was also noted as a problem for rate-based mapping by one participant (P4).

One thing that was surprising to us was that – despite its similarity to mouse-based manipulation of content – the hybrid positional+rate mapping was criticized extensively by participants. Participants found the mode switching challenging to control because of the change in behaviour (P2, P4, P5, P8, P10). In hindsight, this is perhaps unsurprising, as

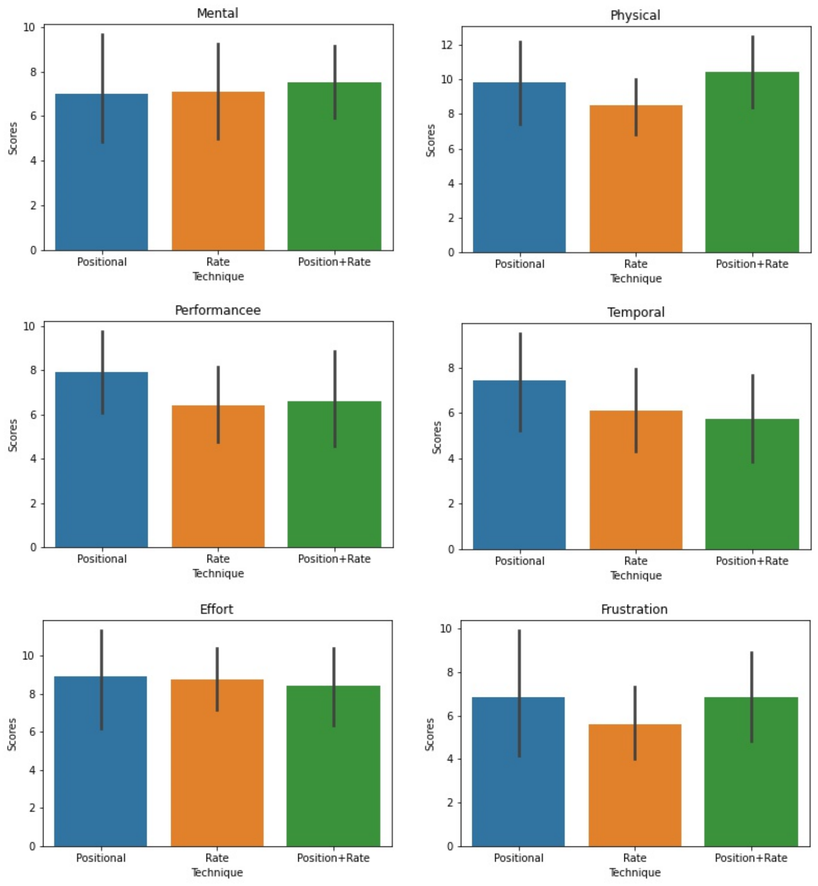


Figure 4.11: NASA-TLX Component Scores, all not significant.

researchers have noted similar problems with mouse-based scrolling in graphical interfaces [18].

4.5 Discussion

The overall task completion time is similar across the three mappings, with positional mapping only performing better when all targets are contained on a single screen. In contrast, participants in our study almost uniformly preferred rate-based mapping. The inconsistencies of the hybrid mapping was identified as problematic from a consistency perspective.

Considering implications for design, we see advantages for both positional and rate-based mappings. First, in our virtual observations of participants, positional mapping caused few problems when scrolling was not required. However, our experiment mixed both non-scrolling and scrolling trials, which, undoubtedly, highlighted the advantage of rate-based control. In an interface where scrolling is unnecessary, for example a video control interface, positional mapping might feel much less tiring and awkward for participants, versus interfaces where large target sets must be navigated, requiring distant reaching and clutching. As a result, combining both quantitative and qualitative measures, a naive interpretation might be to consider positional mappings in interfaces without scrolling and rate-based mappings with scrolling, but we would caution against implementing this approach blindly, primarily because of participants reaction to our hybrid mapping.

In hybrid mapping, while, theoretically, the advantages of positional and rate-based mapping are combined, there are significant challenges presented to participants due to the need to switch modes between positional and rate-based mappings. Inconsistent interface behaviour, and, particularly, the need to control and learn two different mappings within a single pointing task seems problematic and echoes the challenges that users face in mouse-based input when scrolling is required. Issues of mode switch at edges of the display and issues of mode switch and speed control all present greater complexity of learning for the user.

Finally, we note that early in our study we eliminated discrete mappings because of the fatiguing nature of the input and the slow time to acquire targets. From piloting, it was obvious that performance would be much worse than other techniques based upon dependent measures we aimed to collect. However, in environments where only a very small number of targets are presented to the user, it is possible that discrete mappings may still merit exploration. Error rates are, by nature, low with these mappings. Furthermore, there may be a feed-forward [4] benefit to these mappings, because one challenge with mid-air inputs is the twin challenges of discoverability and learnability, i.e. determining what can be done and how to do it. With discrete mappings, when we acquire a user's hand, we can show a remote control on the screen and simply allow users to navigate the remote control, a natural feed-forward mechanism to guide users to commands. This provides a menuing system for input that users already recognize, which could be a benefit for early adoption. Essentially, we would argue, as a caveat to this research, that optimality in speed and/or comfort are not the only factors to consider in the design of mid-air systems. Navigating a virtual remote makes menu options (picture menus, input settings, other smart display commands) immediately visible to participants – or at least as obvious as these commands are when using traditional remotes.

4.5.1 Limitations and Trade-offs

Like any study, ours presents limitations. Obvious examples include the somewhat artificial interface, mimicking but not identical to a video on demand interface, participant numbers, and the non-laboratory environment in which the study was conducted. In terms of interface, we did weigh other alternatives, including traditional Fitts-style interfaces for pointing, but rejected them quite quickly because Fitts-style interfaces neglect issues of scrolling to off-screen targets and are poorly configured for object pointing due to the regularized arrangement of targets that poorly supports intervening targets and object pointing.

Participant numbers are always a challenge in times of Covid. Additional participants might highlight some significant differences in temporal performance. However, given that positional, even qualitatively, outperforms rate-based mapping but is not preferred, we do believe that the more interesting data point is the contradiction between positional and rate-based mapping. This contradiction surprised us, and, given the introduction of mid-air control for targeting on smart displays, seems a useful addition to the literature.

Alongside participant numbers, our participant population was drawn from a relatively narrow demographic, i.e. university-aged students at our university, and was unbalanced by gender. The use of university-aged participants from our institution resulted from constraints placed upon us by the pragmatics of our study and the Covid-19 pandemic. Younger participants (under 40 years of age) are significantly less likely to experience health side-effects from Covid-19, minimizing risk. Furthermore, due to Office of Research Ethics requirements, in-person studies were not permitted, so we designed a study where research equipment was transportable, easily configurable, and thus supported remote experimentation in participants' homes. Because of this need to transfer equipment to participants, our study was limited geographically to the region near our university and, despite campus closures, it remains primarily populated by university students. Too, it is possible that the remote design may also have impacted the gender balance of our participant sample. Individuals who identified as men more frequently volunteered to participate, and this may have been because participants needed to either meet the researcher in a public space or receive equipment at their home during a period when campus was closed and public spaces were less populated due to the pandemic. We assured participants that meetings could take place in public spaces and that they did not need to share their home address. However, as a community and as a society, much work remains to be done in ensuring that all members of our society feel equally safe and comfortable. Regardless of why these factors arose, they do represent risks to generalizability of our results.

The experiment design, itself, also has limitations. As a simple example, our 15.6-

inch display, provided to participants, is both smaller and closer than a typical television. Our goal in providing equipment was to increase control and to limit risk to participants of installing software on their own personal devices (ethics review at our institution is reluctant to allow installation of software on participants' personal devices due to privacy and/or security risks). This drove the need to provide our own hardware to participants in a responsible manner given the context of a global pandemic. However, this fixed hardware represents a risk to generalizability. Specifically, we specified distances for our study to accommodate for the smaller (than typical smart tv) portable display size with closer distance. A 15-inch display at 1 to 1.5m distance, 0.25 radians of visual arc, is similar in visual arc to a 32 inch at 2 to 3m or a 65-inch at 4 to 6m. However, small and or close displays may alter aspects of user behaviour in unanticipated ways, even if the displays appear visually similar in scale.

The gesture recognition system could also be improved. The select gesture has a relatively high error rate, which makes the discrete mapping harder than the other mappings. Provided a better recognition of select gesture, discrete mapping might be able to compare with the others given it's similarity to the traditional remote.

Finally, non-laboratory environments present challenges to the research community when interpreting data and we are mindful of this challenge. While one could argue that there is an ecological benefit to experimentation out of the lab, one benefit of the lab environment is that the environment is optimal. Numerical values collected can be assumed to be a near "best-case" for interactions. This means that, with a similar interface or with published data sets, researchers can replicate the experiment and leverage data and our results to validate their set-up. Real-world contexts for experimentation can preserve internal validity of the data, but they complicate direct comparison between new research and research collected in less controlled, non-laboratory spaces.

Chapter 5

Discussion And Conclusion

5.1 Future Work

This thesis focuses on the object based pointing due to the paradigm on smart TVs. However, cursor based, which is quite common in other devices like computers isn't explored. We could compare those two strategies in the future. Also, as smart devices and connected objects are getting deployed in the world, how gesture interfaces to multiple smart objects within an environment can be supported is also an active area of research.

5.2 Conclusion

This thesis explored two common tasks in TVs using barehand control, including video timeline control and object pointing. We first examines the effectiveness of pointer acceleration in barehand, distant, video timeline control. We evaluate three CD gain variants: a constant function (no pointer acceleration, analogous to direct manipulation) and two different gain functions implementing pointer acceleration, a continuous linear function and a generalised logistic regression function. We find overall benefits to non-constant CD Gain in speed, error, and precision, particularly for our linear CD Gain function. Next, we explore object-pointing mappings for mid-air barehand input to SmartTV displays. Examining three different mappings – discrete, positional, and rate-based – highlight the complementary advantages the different mappings possess. Specifically, we note that discrete mappings may allow easier guidance and greater familiarity, particularly if discrete mappings are used to control a virtual on-screen remote. We note the performance advantages of positional mapping in single-screen displays where scrolling to off-screen targets is not required. Finally, we highlight participants preferences for rate-based control due to the lower physical effort, but also note that it is the mapping most challenge for participants to master. Together these results provide guidance for the design of object-pointing mappings for use in mid-air distant smart display input.

References

[1] Anand Agarawala and Ravin Balakrishnan. *Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen*, page 1283–1292. Association for Computing Machinery, New York, NY, USA, 2006.

[2] Takeshi Asano, Ehud Sharlin, Yoshifumi Kitamura, Kazuki Takashima, and Fumio Kishino. Predictive interaction using the delphian desktop. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, UIST '05, page 133–141, New York, NY, USA, 2005. Association for Computing Machinery.

[3] Ravin Balakrishnan. "beating" fitts' law: virtual enhancements for pointing facilitation. *International Journal of Human-Computer Studies*, 61(6):857 – 874, 2004. Fitts' law 50 years later: applications and contributions from human-computer interaction.

[4] Olivier Bau and Wendy E. Mackay. Octopocus: A dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, page 37–46, New York, NY, USA, 2008. Association for Computing Machinery.

[5] R. Bolt. "put-that-there": Voice and gesture at the graphics interface. In *SIGGRAPH '80*, 1980.

[6] Eugenie Brasier, Olivier Chapuis, Nicolas Ferey, Jeanne Vezien, and Caroline Appert. Arpads: Mid-air indirect input for augmented reality. In *Proceedings of the International Symposium on Mixed and Augmented Reality*, pages 13–pages. IEEE, 2020.

[7] Géry Casiez and Nicolas Roussel. No more bricolage! methods and tools to characterize, replicate and compare pointing transfer functions. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, page 603–614, New York, NY, USA, 2011. Association for Computing Machinery.

[8] Géry Casiez, Daniel Vogel, Ravin Balakrishnan, and Andy Cockburn. The impact of control-display gain on user performance in pointing tasks. *Human–Computer Interaction*, 23(3):215–250, 2008.

[9] Kai-Yin Cheng, Sheng-Jie Luo, Bing-Yu Chen, and Hao-Hua Chu. Smartplayer: User-centric video fast-forwarding. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, page 789–798, New York, NY, USA, 2009. Association for Computing Machinery.

[10] Andy Cockburn and Joshua Savage. Comparing speed-dependent automatic zooming with traditional scroll, pan and zoom methods. In Eamonn O'Neill, Philippe Palanque, and Peter Johnson, editors, *People and Computers XVII — Designing for Society*, pages 87–102, London, 2004. Springer London.

[11] Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowitcz, Derek Nowrouzezahrai, Ravin Balakrishnan, and Karan Singh. Video browsing by direct manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, page 237–246, New York, NY, USA, 2008. Association for Computing Machinery.

[12] Clifton Forlines, Daniel Vogel, and Ravin Balakrishnan. Hybridpointing: Fluid switching between absolute and relative pointing with a direct input device. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, page 211–220, New York, NY, USA, 2006. Association for Computing Machinery.

[13] Celeste Groenewald, Craig Anslow, Junayed Islam, Chris Rooney, Peter J Passmore, and BL Wong. Understanding 3d mid-air hand gestures with interactive surfaces and displays: a systematic literature review. 2016.

[14] Tovi Grossman and Ravin Balakrishnan. The bubble cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, page 281–290, New York, NY, USA, 2005. Association for Computing Machinery.

[15] Yves Guiard, Renaud Blanch, and Michel Beaudouin-Lafon. Object pointing: a complement to bitmap pointing in guis. In *Proceedings of Graphics Interface 2004*, pages 9–16. Citeseer, 2004.

[16] Juan David Hincapié-Ramos, Xiang Guo, Paymahn Moghadasian, and Pourang Irani. Consumed endurance: a metric to quantify arm fatigue of mid-air interactions. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2014.

[17] Wolfgang Hürst and Konrad Meier. Interfaces for timeline-based mobile video browsing. In *Proceedings of the 16th ACM International Conference on Multimedia*, MM '08, page 469–478, New York, NY, USA, 2008. Association for Computing Machinery.

[18] Takeo Igarashi and Ken Hinckley. Speed-dependent automatic zooming for browsing large documents. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, page 139–148, New York, NY, USA, 2000. Association for Computing Machinery.

[19] Herbert D. Jellinek and Stuart K. Card. Powermice and user performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, page 213–220, New York, NY, USA, 1990. Association for Computing Machinery.

[20] Thorsten Karrer, Malte Weiss, Eric Lee, and Jan Borchers. Dragon: A direct manipulation interface for frame-accurate in-scene video navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, page 247–250, New York, NY, USA, 2008. Association for Computing Machinery.

[21] Keiko Katsuragawa, Krzysztof Pietroszek, James R Wallace, and Edward Lank. Watchpoint: Freehand pointing with a smartwatch in a ubiquitous display environment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 128–135, 2016.

[22] D. Kimber, T. Dunnigan, A. Girgensohn, F. Shipman, T. Turner, and T. Yang. Trailblazing: Video playback control by direct object manipulation. In *2007 IEEE International Conference on Multimedia and Expo*, pages 1015–1018, 2007.

[23] Panayiotis Koutsabasis and Panagiotis Vogiatzidakis. Empirical research in mid-air interaction: A systematic review. *International Journal of Human–Computer Interaction*, 35(18):1747–1768, 2019.

[24] Mingyu Liu, Mathieu Nancel, and Daniel Vogel. Gunslinger: Subtle arms-down mid-air interaction. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 63–71, 2015.

[25] I. MacKenzie. Fitts' law as a performance model in human-computer interaction. 1992.

[26] Justin Matejka, Tovi Grossman, and George Fitzmaurice. *Swifter: Improved Online Video Scrubbing*, page 1159–1168. Association for Computing Machinery, New York, NY, USA, 2013.

[27] Fabrice Matulic and Daniel Vogel. Multiray: multi-finger raycasting for large displays. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.

[28] Sven Mayer, Valentin Schwind, Robin Schweigert, and Niels Henze. The effect of offset correction and cursor on mid-air pointing in real and virtual environments. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.

[29] Michael McGuffin and Ravin Balakrishnan. Acquisition of expanding targets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, page 57–64, New York, NY, USA, 2002. Association for Computing Machinery.

[30] Jörg Müller. Dynamics of pointing with pointer acceleration. In *IFIP Conference on Human-Computer Interaction*, pages 475–495. Springer, 2017.

[31] Mathieu Nancel, Olivier Chapuis, Emmanuel Pietriga, Xing-Dong Yang, Pourang P. Irani, and Michel Beaudouin-Lafon. High-precision pointing on large wall displays using small handheld devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, page 831–840, New York, NY, USA, 2013. Association for Computing Machinery.

[32] Mathieu Nancel, Emmanuel Pietriga, Olivier Chapuis, and Michel Beaudouin-Lafon. Mid-air pointing on ultra-walls. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 22(5):1–62, 2015.

[33] Krzysztof Pietroszek, Liudmila Tahai, James R Wallace, and Edward Lank. Watchcasting: Freehand 3d interaction with off-the-shelf smartwatch. In *2017 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 172–175. IEEE, 2017.

[34] Suporn Pongnumkul, Jue Wang, Gonzalo Ramos, and Michael Cohen. Content-aware dynamic timeline for video browsing. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, page 139–142, New York, NY, USA, 2010. Association for Computing Machinery.

[35] Klaus Schoeffmann, Marco A. Hudelist, and Jochen Huber. Video interaction tools: A survey of recent work. *ACM Comput. Surv.*, 48(1), September 2015.

[36] Shaishav Siddhpuria, Sylvain Malacria, Mathieu Nancel, and Edward Lank. Pointing at a distance with everyday smart devices. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–11, 2018.

[37] Rafael Veras, Gaganpreet Singh, Farzin Farhadi-Niaki, Ritesh Udhani, Parth Pradeep Patekar, Wei Zhou, Pourang Irani, and Wei Li. *Elbow-Anchored Interaction: Designing Restful Mid-Air Input.* Association for Computing Machinery, New York, NY, USA, 2021.

[38] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. Mediapipe hands: On-device real-time hand tracking, 2020.

[39] Futian Zhang, Sachi Mizobuchi, Wei Zhou, Taslim Arefin Khan, Wei Li, and Edward Lank. Cd gain for barehand timeline browsing. In *Proceedings of Interact 2021*, page to appear, 2021.