Leveraging CD Gain for Precise Barehand Video Timeline Browsing on Smart Displays

Futian Zhang[1(✉)], Sachi Mizobuchi[2], Wei Zhou[2], Taslim Arefin Khan[2], Wei Li[2], and Edward Lank[1]

[1] Cheriton School of Computer Science, University of Waterloo, Waterloo, Canada
futian.zhang@uwaterloo.ca
[2] Human-Machine Interaction Lab, Huawei, Toronto, Canada

Abstract. One common task when controlling smart displays is the manipulation of video timelines. Given current examples of smart displays that support distant bare hand control, in this paper we explore CD Gain functions to support both seeking and scrubbing tasks. Through a series of experiments, we demonstrate that a linear CD Gain function provides performance advantages when compared to 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 (±5 s) is less than one third the error of a Constant gain.

Keywords: CD Gain · Gesture control · Smart TV · Video browsing

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. 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.

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

Published by Springer Nature Switzerland AG 2021
C. Ardito et al. (Eds.): INTERACT 2021, LNCS 12935, pp. 72–91, 2021.
https://doi.org/10.1007/978-3-030-85610-6_5

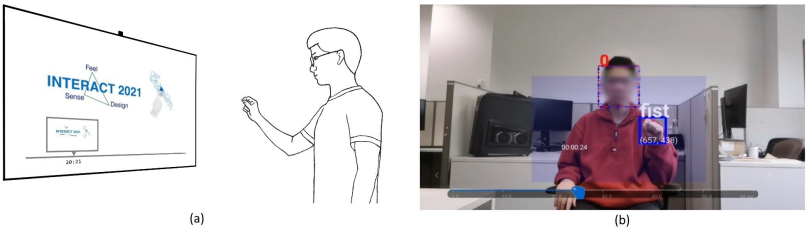


Fig. 1. Video timeline control for smart TVs. (a) The usage in a real world task. (b) The view of the camera on the smart TV when the user is navigating.

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 s), 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 [4,5,10,12,22]. 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.

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.

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 [7,8,17]), as Balakrishnan [1] 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 [19] 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 paper, 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 [19]. 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 [2] and significant recent research in selecting CD gain functions [19], 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, in this paper we present 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.

2 Related Work

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 [4] and Pongnumkul et al.'s [22] 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 [5,10,12,14].

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 [23].

2.2 Mid-Air Pointing

Koutsabasis et al. [13] provided a systematic review on mid-air interaction. Due to the inaccuracy of mid-air pointing, a significant body of work in this area focuses on increasing pointing precision. For example, Nancel et al. [20] 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. [16] 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 [6]) by alternating between direct or constant CD Gain and indirect manipulation.

Alongside barehand input, a significant body of work uses smartwatches or smartphones [11,21,24] to perform mid-air pointing on large displays, including the incorporation of linear or discontinuous CD Gain functions [11]. Finally, techniques such as Multiray [15] employed multiple fingers to raycast on large displays and the system creates patterns for different actions.

2.3 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. [3] 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 [9].

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 [2], 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. [19] 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 specification 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 [2]. 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.

3 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 paper, 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 setup, 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 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).

Next, our algorithm specifies a rectangular gesture input area below the face (see Fig. 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 Fig. 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.

Fig. 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, Eq. 1 shows the relationship between on-screen translation, ΔT, and in-air movement of the hand, Δx.

$$\Delta \mathrm{T} = \Delta x \cdot cdgain \tag{1}$$

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.

4 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.

4.1 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.

4.2 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 in. 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-in. display.

To control the timeline, participants were asked to perform a "pinch open" gesture (Fig. 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 7200 s, with a scale every 600 s) 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 (Fig. 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 10 s-equivalent of movement of the cursor on the 7200 s-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 10 s from trial start. In both cases, a audible confirmation sound was played and the trial ended.

4.3 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

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

participant (Tasks 1 and 2 were fully counter-balanced; task 3 was always the final task performed).

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 [3] for a review). As well, some past research has used non-continuous functions as CD Gain functions. For example Muller [18] used a minimum CD Gain of 3.0 for mouse speed up to 10 cm/s (i.e. constant gain); a maximum gain of 5.0 for mouse speed above 30 cm/s (i.e. a second constant gain), and a linear function for mouse speed between 10 and 30 cm/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 calculated 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 (Eq. 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.

$$[h]Y(x) = A + \frac{K - A}{(C + Qe^{-B(x-M)})^{\frac{1}{v}}} \tag{2}$$

Our choice of a generalised logistic function was influenced by pointer acceleration curves used in modern desktop operating systems [2] and by recent research in pointer acceleration dynamics [18]. The GLF both replicates very closely the form of commercial CD Gain functions and serves as an elegant, continuous approximation of Muller's [18] piecewise function. We posit that the CD Gain functions we select have significant benefits in terms of reproducibility. As Casiez and Roussel note, commercial CD Gain functions are difficult to reproduce exactly [2], whereas ours can easily be replicated using the above information. It is possible that a piecewise function with linear range and max-min discontinuities [18] 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 visualises CD gain by hand speed in each CD gain condition.

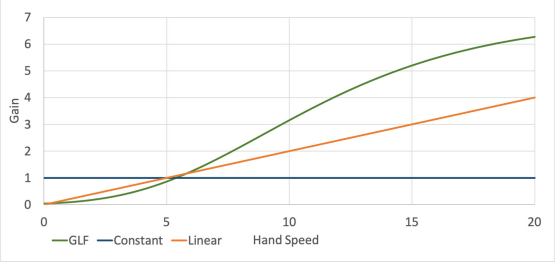


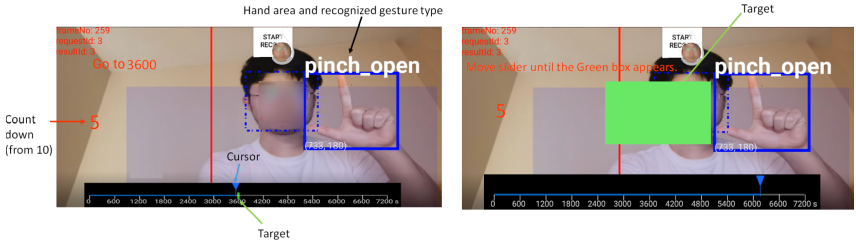
Fig. 3. CD gain conditions

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 s) point on a timeline with 7200 divisions (analogous to 1 s targeting in a two-hour video).

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 timeline, and the participants were instructed to move the cursor within the target range as quickly and as accurately as possible (Fig. 4a).

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 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 (Fig. 4b) (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.



(a) Task 1 (b) Task 2

Fig. 4. Experimental conditions. (Color figure online)

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.

4.4 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 1–1.5 m 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 $0.3\,\mathrm{m} = 0.5\,\mathrm{m}$ 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 s 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 h, 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.

4.5 Data Collection

Considering all factors above, each participant completed 351 trials: ((3 distances × 3 tolerances × 6 blocks × 2 tasks × 3 CD gain conditions) + (9 trials × 1 task × 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 s. Likert question data was stored on the smartphone and downloaded after each participant.

5 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.

5.1 Seeking

Accuracy. Figure 5 (left) shows the error rate (the proportion of trials participants could not acquire the target within 10 s) 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 5 (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 ($t_{11} = -3.24$, $p < .05$).

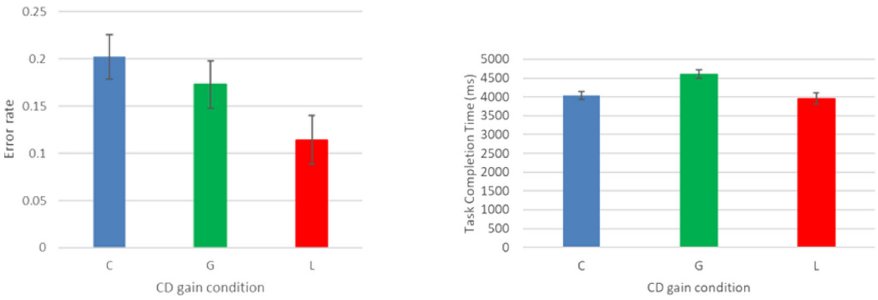


Fig. 5. Mean error rate and Task completion time by CD gain in Seeking Task

5.2 Scrubbing

Accuracy. Figure 6 (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 6 (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 ($t_{11} = -3.24$, $p < .05$).

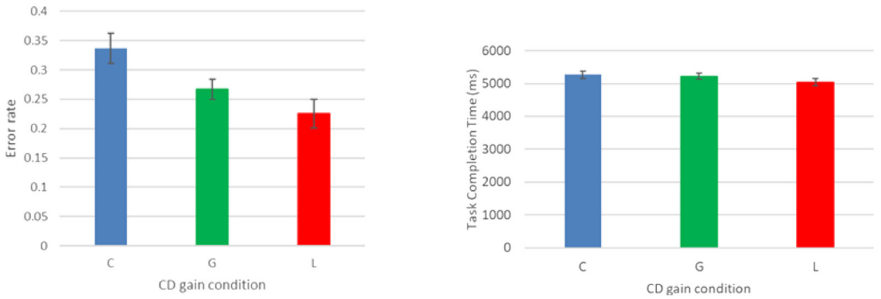


Fig. 6. Mean error rate and Task completion time by CD gain in Scrubbing Task.

Subjective Ratings. Figure 7 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.

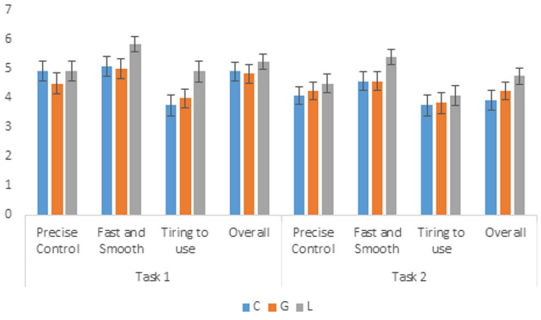


Fig. 7. Mean subjective ratings on CD gain conditions with standard errors

5.3 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 s). We note that, because this involves sub-pixel selection on a 4K resolution display, it is an exceedingly difficult task. Selecting a 1-s 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-s interval using a mouse along a timeline is also exceedingly difficult. A more important question is *how closely* a 1-s 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 ($t_{11} = 6.44$, $p < .001$).

6 Discussion

In our introduction, we posed two primary questions: 1) What CD Gain function is most effective in bare-hand 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 [3]. 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 [19]; 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 [1,7,17] 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.

6.1 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 s-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-in. display at 1 to 1.5 m distance, 0.25 rad of visual arc, is similar in visual arc to a 32 in. at 2 to 3 m or a 65-in. at 4 to 6 m. 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 setup was measured at 136 ms. Approximately 2/3 of this latency (90 ms) is from image processing and movement mapping in Android, 20 ms is due to USB-C communication, monitor response (4 ms) and refresh (60 Hz), and the remainder

is due to camera capture latency from the Mate30 Pro front camera (30 fps). 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.

Remote Experiments: Advantages and Disadvantages. The remote nature of this experiment presents both advantages and disadvantages. From the perspective of advantages, we believe that conducting an experiment in a home-based environment for home-based technologies (e.g. Smart TVs) increases ecological validity versus a laboratory environment. Also, while not availed of in our study, remote experimentation can increase access to studies and increase the demographic heterogeneity of participants because participants who may otherwise not have access to laboratory-based settings may still be able to participate.

On the other hand, there are disadvantages, particularly in logistics and replicability. Logistically, diagnosing problems in remote settings is challenging. If code or tracking performs poorly, it can be difficult to ascertain if it is due to participants' positioning, back lighting, front lighting, or some other factor. We experienced this in our own testing configurations: as one example, side lighting was too bright in one author's desk area, requiring him to close the blinds. This made the environment too dark, which he solved by placing a bright white background on his display screen to increase front lighting. Asking participants to reposition experimental equipment or furniture, change angles, and perform other corrections represents a challenge for those participants. We were fortunate that our participants were able and willing to assist us in configuring their environment during the experiment.

Replicability costs are also increased, both for us and more generally for the research community. In laboratory environments, one can assume optimal experimental conditions. This means that, if another researcher wishes to use a lab-based data set, then they can assume optimality. If their proposed intervention performs better than the lab-based dataset, this represents with high likelihood a real improvement. Our measured dependent variables are likely suboptimal compared to a laboratory setting, meaning that direct numerical comparison to our data must be performed cautiously. This does not impact the internal validity of our results – factors are constant across conditions in our study – but it means that experiments wishing to contrast with ours may need to fully replicate the study rather than relying on an open data set. Essentially, experimental contrasts may be dependent on a repeated measures design to control both participant and environmental variability simultaneously when running

experiments. We would argue that this is generally true of remote experiments, and if between subjects contrasts are desired in remote experimentation, it may be necessary to ensure a very high number of participants to obtain sufficient statistical power.

Overall, our goal during remote experimentation was to attempt to replicate as closely as possible a controlled environment despite the varied locations (hence the set-up constraints, equipment exchange, and video-based presence of a member of the research team). It is unclear how much this differs from a lab context (versus in-the-wild). We do identify statistical differences via appropriate analyses, and we believe these differences would hold in-lab. Whether the differences would remain as significant in a less constrained in-home environment with poor lighting, messy rooms, multiple people on camera, oblique angles, varied postures, etc. is unclear.

7 Conclusion

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.

Acknowledgements. We thank participants and our Office of Research Ethics for valuable assistance during the design of our remote study. This research was funded by a grant from Waterloo-Huawei Joint Innovation Laboratory.

References

1. Balakrishnan, R.: "Beating" Fitts' law: virtual enhancements for pointing facilitation. Int. J. Hum.-Comput. Stud. **61**(6), 857–874 (2004). https://doi.org/10.1016/j.ijhcs.2004.09.002. http://www.sciencedirect.com/science/article/pii/S107158190400103X. Fitts' law 50 years later: applications and contributions from human-computer interaction
2. Casiez, G., Roussel, N.: 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 2011, pp. 603–614. Association for Computing Machinery, New York (2011). https://doi.org/10.1145/2047196.2047276
3. Casiez, G., Vogel, D., Balakrishnan, R., Cockburn, A.: The impact of control-display gain on user performance in pointing tasks. Hum.-Comput. Interact. **23**(3), 215–250 (2008). https://doi.org/10.1080/07370020802278163. https://www.tandfonline.com/doi/abs/10.1080/07370020802278163

4. Cheng, K.Y., Luo, S.J., Chen, B.Y., Chu, H.H.: Smartplayer: user-centric video fast-forwarding. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2009, pp. 789–798. Association for Computing Machinery, New York (2009). https://doi.org/10.1145/1518701.1518823
5. Dragicevic, P., Ramos, G., Bibliowitcz, J., Nowrouzezahrai, D., Balakrishnan, R., Singh, K.: Video browsing by direct manipulation. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2008, pp. 237–246. Association for Computing Machinery, New York (2008). https://doi.org/10.1145/1357054.1357096
6. Forlines, C., Vogel, D., Balakrishnan, R.: 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 2006, pp. 211–220. Association for Computing Machinery, New York (2006). https://doi.org/10.1145/1166253.1166286
7. Grossman, T., Balakrishnan, R.: 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 2005, pp. 281–290. Association for Computing Machinery, New York (2005). https://doi.org/10.1145/1054972.1055012
8. Guiard, Y., Blanch, R., Beaudouin-Lafon, M.: Object pointing: a complement to bitmap pointing in GUIs. In: Proceedings of Graphics Interface 2004, pp. 9–16. Citeseer (2004)
9. Jellinek, H.D., Card, S.K.: Powermice and user performance. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 1990, pp. 213–220. Association for Computing Machinery, New York (1990). https://doi.org/10.1145/97243.97276
10. Karrer, T., Weiss, M., Lee, E., Borchers, J.: 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 2008, pp. 247–250. Association for Computing Machinery, New York (2008). https://doi.org/10.1145/1357054.1357097
11. Katsuragawa, K., Pietroszek, K., Wallace, J.R., Lank, E.: Watchpoint: freehand pointing with a smartwatch in a ubiquitous display environment. In: Proceedings of the International Working Conference on Advanced Visual Interfaces, pp. 128–135 (2016)
12. Kimber, D., Dunnigan, T., Girgensohn, A., Shipman, F., Turner, T., Yang, T.: Trailblazing: video playback control by direct object manipulation. In: 2007 IEEE International Conference on Multimedia and Expo, pp. 1015–1018 (2007)
13. Koutsabasis, P., Vogiatzidakis, P.: Empirical research in mid-air interaction: a systematic review. Int. J. Hum.-Comput. Interact. **35**(18), 1747–1768 (2019)
14. Matejka, J., Grossman, T., Fitzmaurice, G.: Swifter: Improved Online Video Scrubbing, pp. 1159–1168. Association for Computing Machinery, New York (2013). https://doi.org/10.1145/2470654.2466149
15. Matulic, F., Vogel, D.: Multiray: multi-finger raycasting for large displays. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, pp. 1–13 (2018)
16. Mayer, S., Schwind, V., Schweigert, R., Henze, N.: 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, pp. 1–13 (2018)

17. McGuffin, M., Balakrishnan, R.: Acquisition of expanding targets. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2002, pp. 57–64. Association for Computing Machinery, New York (2002). https://doi.org/10.1145/503376.503388
18. Müller, J.: Dynamics of pointing with pointer acceleration. In: Bernhaupt, R., Dalvi, G., Joshi, A., Balkrishan, D.K., O'Neill, J., Winckler, M. (eds.) INTERACT 2017. LNCS, vol. 10515, pp. 475–495. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67687-6_33
19. Nancel, M., Chapuis, O., Pietriga, E., Yang, X.D., Irani, P.P., Beaudouin-Lafon, M.: High-precision pointing on large wall displays using small handheld devices. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2013, pp. 831–840. Association for Computing Machinery, New York (2013). https://doi.org/10.1145/2470654.2470773
20. Nancel, M., Pietriga, E., Chapuis, O., Beaudouin-Lafon, M.: Mid-air pointing on ultra-walls. ACM Trans. Comput.-Hum. Interact. (TOCHI) **22**(5), 1–62 (2015)
21. Pietroszek, K., Tahai, L., Wallace, J.R., Lank, E.: Watchcasting: freehand 3D interaction with off-the-shelf smartwatch. In: 2017 IEEE Symposium on 3D User Interfaces (3DUI), pp. 172–175. IEEE (2017)
22. Pongnumkul, S., Wang, J., Ramos, G., Cohen, M.: Content-aware dynamic timeline for video browsing. In: Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology, UIST 2010, pp. 139–142. Association for Computing Machinery, New York (2010). https://doi.org/10.1145/1866029.1866053
23. Schoeffmann, K., Hudelist, M.A., Huber, J.: Video interaction tools: a survey of recent work. ACM Comput. Surv. **48**(1) (2015). https://doi.org/10.1145/2808796
24. Siddhpuria, S., Malacria, S., Nancel, M., Lank, E.: Pointing at a distance with everyday smart devices. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, pp. 1–11 (2018)