EasyPZ: Comparing Pan and Zoom Interactions on Mobile and Desktop with 1D Timelines

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

ABSTRACT

Pan and zoom interaction allow us to explore multi-scale data on fixed-size displays; yet, there are few surveys to date which compare interaction technique compatibility and performance. We review existing pan and zoom techniques and identify the characteristics which define their full or partial compatibility. In our EASYPZ open source JavaScript library, we provide implementations for 12 interchangable pan and zoom techniques. Using timelines as our test scenario, we conducted a study to measure the performance of six common 1D pan and zoom interaction pairs, across horizontal and vertical orientations, and mobile and desktop devices. Analysis of these results leads to recommendations for application developers, with our results and software providing a community foundation for continued pan and zoom interaction research.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

Author Keywords

Pan; zoom; multi-scale; mobile; desktop; timelines; visualization; interaction study.

INTRODUCTION

Pan and zoom interaction allow us to explore multi-scale data on fixed-size displays (Cockburn et al. provide an overview [4]). The human-computer interaction literature contains many different techniques for pan, zoom in, and zoom out, and these must be mapped to an input device to create an interaction. However, only some techniques are complementary given a particular input device. For instance, the common motions of rotating a mouse wheel or swiping with a finger can be mapped either to pan or zoom, often in combination with click/tap mode switches and gestures.

This interaction design space is large, including double-click zoom, hold zoom, drag or dynamic zoom, brush or marquee zoom, and rub zoom, along with behavior parameters for each.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

Choosing an appropriate design is difficult and, due to the wide application of pan and zoom, even a small performance difference would have a large collective impact. Yet, there are few surveys with an overview of technique compatibility or performance, and there is no set of consistent implementations or experiments shared by the community upon which to test potential improvements. Furthermore, while many technique publications include user studies, they are often small and in controlled conditions. This does not expose real-world device and participant behavior, which is often necessary to address design questions which affect a wide variety of display and interaction device combinations.

The problem of pan and zoom interaction design is made worse in two ways. First, mobile devices are now the predominant computing platform. Many existing studies were performed on desktops, but small screens increase the need for multi-scale interaction while also lowering the navigation bandwidth [7]. Second, as our ability to capture, store, and distribute multi-scale data increases, so too does the need to develop and compare new interaction methods. It is now expected that we can find 'that one photo we remember' from an approximate time within a personal digital photo collection spanning tens of thousands of images and 20 years (6×10^8 seconds); or finding a specific frame of an event at an approximate time in a home security video (10 hours @ 30 fps = 10^6 frames) [21].

Which interaction techniques are effective? This paper begins to address these questions:

1. We review pan and zoom techniques and tabulate common technique compatibility for mobile and desktop devices.

2. We compare pan and zoom techniques in a multi-scale pointing task on 1D timelines in a three-part study conducted on Amazon's Mechanical Turk with 318 participants, across both mobile and desktop devices. First, we compare number lines and timelines, and discover that timelines are faster to navigate than number lines. Second, we compare horizontal and vertical timelines, and discover that vertical can be faster than horizontal on mobile, but not on desktop. Third, we compare six common pan and zoom interaction combinations, and state how measurement time varies across participants, devices, and target difficulty. Then, we discuss the design implications of our results across mobile and desktop.

3. We release an open source Javascript library—EASYPZ—with 12 interaction techniques plus variants. These are

decoupled from any visualization and work in 1D or 2D. We also provide a library with ready-to-use 1D number lines and timelines—EASYPZLINES. Along with this, we provide a Website—*URL withheld for anonymity*—which allows researchers to interactively compare pan and zoom techniques and observe our associated study data, including replays of study trials for behavior analysis.

The scope of pan and zoom interactions is large, and we must limit our study. First, we use 1D pan and zoom rather than 2D (e.g., maps), which allows us to explore '2D for 1D' interaction designs. Many of our studied techniques have intuitive 2D equivalents, and our EASYPZ library allows 2D mapping. Second, we consider single timelines only, and not multi-timeline interfaces [25]. Third, we do not consider scrollbar designs: they are uncommon on mobile, and on desktop they have less frequent use given explicit pan inputs (scroll wheels) and gestures (two-finger scroll). Further, for this kind of real-world study, we only have loose control over which devices are used, from which we can see how interaction design affects the participant body as a whole. Thus, 'mobile' refers to smartphones, and 'desktop' refers to any larger-screened device such as a laptop or workstation.

That said, our work tests a broad set of pan and zoom interactions prompted by the common timeline scenario. Our open source EASYPZ library, comparison Website with referenced user study data, and design recommendations will help developers make decisions about how best to implement pan and zoom. Further, these tools will help researchers compare and contrast new pan and zoom techniques.

RELATED WORK

Any kind of linear sensing device can be used as a physical interface for panning or zooming. We choose to include the most common devices for mobile (touchscreens) and desktop (mice, scroll wheels, touchpads). Given sensor readings, and in combination with buttons to provide mode switches, we must devise a scheme which provides pan, zoom in, and zoom out functions via a scale-space mapping or transfer function. This function can be linear or non-linear, it can imitate physical phenomena such as rigid body transformations and momentum or elasticity, it can exist in dimensions larger than the data (e.g., exploiting the second dimension of an input device designed for a 2D display; '2D for 1D'), and it can be based on complex gesture progression. Navigating this design space is a fundamental problem in interaction design for exploring multi-scale data.

Early multi-scale or zoomable interfaces, such as Pad, Pad++, and MuSE [22, 3, 5] defined zoom based on holding a mouse button down or dragging a slider. The interaction design space has been explored through brush or marquee zoom, anchor or dynamic zoom, pinch to zoom [15], elastic interaction for precise manipulation [17], pan-speed-dependent zooming [12], flick pan or zoom [23], and cyclic gestures based on rubbing [20] or circles [16]. Moving beyond 2D pointing devices, the literature explores mid-air gestures [19], Webcam-based zooms for desktop [8] and mobile [13], plus mobile-specific tilt sensors [10] and camera-based device spatial positioning [27]. Typically, each study compares two or three interactions in controlled conditions. We try to tease out how many common interactions compare in real world conditions.

Some interactions require precision control of a slider without content zoom. Ahlberg and Shneiderman ask participants to use a slider to select an item from a list of 10^4 entries [1], and allow orthogonal mouse drags to switch between coarse, medium, and fine pan granularity. Appert and Fekete's OrthoZoom allows continuous granularity control, and find it to be faster than pan-speed-adjusted zooming [2]. One application for 'precision pan' is video, where frame-accurate seeking methods are often required for long videos [21] (Schoeffmann provides an overview [26]). Ramos and Balakrishnan develop position plus velocity sliders, which allow drag motion perpendicular to the slider to control seek granularity [24]. Hurst et al. introduce video timelines with smoothly varying seek pan granularity with orthogonal cursor movement [11]. This slider approach has been adopted in YouTube [6]. These techniques are not principally restricted to pan-only cases, and can be thought of as special cases of drag zoom.

We can relate interaction designs by a theory of scale space interaction. Guiard and Beaudouin-Lafon empirically showed that Fitts' Law can apply to multi-scale pointing [7]. Across indices of difficulty (*ID*) from 5–30 bits, they asked seven participants to move between visual targets using bi-manual control: a game controller throttle for zoom and either a mouse or stylus for pan. They found target pointing measurement time to vary linearly with target ID, with performance variation represented by the slope of the line (e.g., zoom in vs. zoom out). This creates a model for the comparison of different multi-scale interaction techniques.

During interaction, the user must decide how and when to use the constituent pan, zoom in, and zoom out functions. For instance, below 8 bit IDs, the distance is sufficiently small that the user may pan only, but high-bit ID targets require 10+ seconds of interaction with many different combinations of pan, zoom in, and zoom out functions possible [7]. The data itself also affects interaction behavior, as the user must take time to orient the data view to their mental map of the data. This 'idle' time can be affected by interaction design, such as by varying zoom animation parameters to ease the view transition. Both the interaction difficulty and the human data processing can cause 'desert fog' [14], where a user becomes lost in multi-scale space such as by zooming in to an incorrect target. This may require full or partial zoom out for reorientation. On this last point, some designs explicitly provide a 'full zoom out' or 'reset view' button to speed high-ID target pointing. Overall, we analyze participant behavior across interactions to provide insights into the limitations of existing approaches.

STUDY INTERACTION DESIGNS

Among the large design space of interactions, we describe the compatibility between a subset of individual pan and zoom functions. As broad-base interaction design comparisons are rare, we chose to focus on common methods used in popular applications and those which apply both to mobile and desktop devices with 2D pointing devices.

Interaction Method	Pan	Zoom in	Zoom out	Scroll	Double Click (prim.)	Double Click (sec.)	Drag (prim.)	Drag + Back (prim.)	Drag (sec.)	Drag + Back (sec.)	Two Finger Drag
2D Drag Pan	✓	✗	✗				●	○			
2D Flick Pan	✓	✗	✗				●	○			
1D Scroll Pan	✓	✗	✗	●							
2D Two-finger Pan	✓	✗	✗								●
2D 2x Click Zoom (prim.)	✗	✓/✗	✗/✓		●						
2D 2x Click Zoom (sec.)	✗	✓/✗	✗/✓			●					
2D Scroll Zoom	✗	✓	✓	●							
2D Pinch Zoom+Pan	✓	✓	✓								●
1D Dynamic Zoom	✗	✓	✓				○		●	●	
2D Brush Zoom	✗	✓	✗				●	○			
2D Hold to Zoom	✗	✓/✗	✗/✓				○	○			
2D Click + Hold Zoom	✗	✓/✗	✗/✓								
2D Rubbing (prim.)	✗	✓/✗	✗/✓					●			
2D Rubbing (sec.)	✗	✓/✗	✗/✓				○			●	

Interaction Key

Scroll
Double Click (prim.)
Double Click (sec.)
Drag (prim.)
Drag + Back (prim.)
Drag (sec.)
Drag + Back (sec.)
Two Finger Drag

Table 1. Overview of pan and zoom techniques and their compatibility. Methods are identified as working on mobile ( ), and/or desktop( ). 1D methods only work on one-dimensional visualizations such as timelines, while 2D methods can additionally operate on two-dimensional contents such as maps. Methods can support (✓) or not support (✗) panning ( ), zooming in ( ), and zooming out ( ), and some methods can be used for either zooming in or zooming out (✓/✗). Columns 1-10 refer to specific input methods. A ● indicates that an input method is the primary input method for an interaction, while a ○ indicates that an input method is used as additional input for the interaction. Two interactions with ● in the same column are never compatible, while two interactions that have one ● and one ○ or two ○ have limited compatibility. Two interaction methods are fully compatible if there is no overlap between their filled columns.

For pan, we describe the standard desktop dragging with a linear transfer function ('drag pan'), the standard mobile dragging with flick momentum ('flick pan'), using the scroll wheel on a desktop mouse or a scroll gesture on a touchpad ('scroll pan'), and using the touchscreen with a two-finger swipe ('two-finger pan').

For zoom, we describe double click or tap with a primary or secondary button ('2x click zoom'), holding down a button or pressing a touchscreen without releasing the finger ('hold to zoom'), using the scroll wheel on a desktop mouse or a scroll gesture on a touchpad ('scroll zoom'), two-finger pinch to zoom ('pinch zoom'), drag-selecting a rectangular zoom region, often called marquee zoom or brushing ('brush zoom'), dragging along the orthogonal axis for continuous zoom granularity control ('dynamic zoom'), and the cyclic gesture of bi-directional rubbing to zoom in and out ('rub zoom' [20]), e.g., where rubbing horizontally zooms in and rubbing vertically zooms out.

With this subset of interactions, plus their individual parameters, there are many possible (and impossible) combinations to make an effective set of pan, zoom in, and zoom out pan functions. By analyzing these mappings via mode switches, we demonstrate these compatibilities in Table 1. There are many other techniques that could be compared, but this subset covers most common cases and interaction class types. Most techniques have multiple tunable parameters, such as zoom speed, minimum interaction time, minimum or maximum pointer speed during the beginning of the interaction. This affects their compatibility. For example, by requiring a reasonable minimum speed at the beginning of a pan interaction, pan becomes compatible with hold to zoom, which has a maximum speed at the beginning of the interaction. In our third study with participants, we pick six combinations from this table which cover the majority of common pairings.

EasyPZ Software

We implement each of these interaction techniques in an open source JavaScript library. These techniques are independent of any visualization, and have both horizontal and vertical variants, with linear and non-linear mappings, momentum, plus technique-specific variants for dynamic zoom, which has many instances across the literature. These can be used in conjunction with any one or two-dimensional content such as sliders, timelines, or maps.

To freely move between these interactions, a linked visualization should implement three functions: `onPanned(x, y)`, `onZoomed(x, y, scaleChange?, absoluteScaleChange?, targetX?, targetY?)`, and `onSaveCurrentScale()` (which sets a scale 'anchor' for dynamic-type zooms). Application developers can focus on the data and visualization manipulations necessary for scaling the content, while EasyPZ handles the interaction methods used as input for this scale change. As a demonstrator of how to couple EasyPZ to a specific data visualization, we build a supplemental library—EasyPZLines—which provides ready-to-use examples for number lines and timelines.

While some interaction techniques are simple, the nuances of dynamic zoom and rub zoom require significant description. As such, this library also serves as the reference to explain how each of our study techniques were implemented. Along with EasyPZLines, these serve as a source for replication studies and for further research. Our software release also

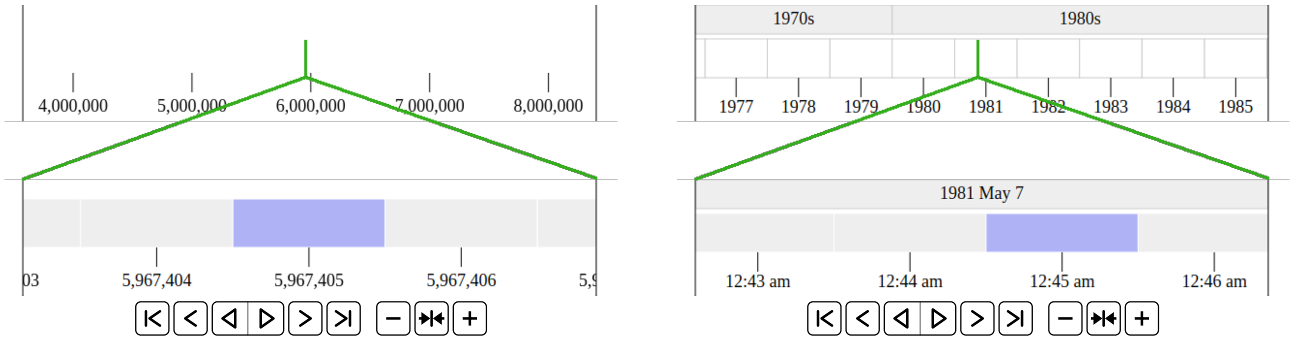


Figure 1. Study 1 conditions. *Left*: Number line. *Right:* Timeline. *Top:* Zoomed out. *Bottom:* Zoomed in, where the green line indicates the zoom. Viewed electronically and with the controls, this figure plays an animation showing an example pan/zoom interaction (supported in Adobe Reader, PDF-XChange, and Foxit Reader).

includes a Webpage—*URL withheld for anonymity*—which allows researchers to interactively compare pan and zoom techniques and observe our associated study data, including replays of study trials for behavior analysis.

USER STUDIES

We wish to measure the real-world performance and variability of different interaction techniques across mobile and desktop. We describe the experimental procedure, the gathered data, and the data analysis of three mechanical turk user studies for designed this task. We use 1D timelines to frame our interaction study with a common real-world interface element (Figure 1).

Our first study addressed the underlying data on a line: "*Are numbers or times easier to navigate?*". This represents an application difference, say, between a video interface indexed by frames or by timecode. Our second study assessed a common interface design decision: "*Is timeline pan/zoom easier horizontally or vertically*". From these two questions, we pick the representation which was most perfomant across participants—vertical timelines—for our third study into interaction design: "*How effective are different pan and zoom techniques?*".

Multi-scale Pointing Task

Each trial in our studies asked participants to find a known target time/date on a pan and zoom line (or a number in study 1). We ask participants to find targets at known positions on the line to stimulate pan and zoom interaction, but in a real application this task might be accomplished more efficiently through text entry or spinners. To investigate performance over multi-scale data, we use a number line spanning from zero to 20 million, which corresponds to all times to the minute over a period of 40 years for the timeline. Upon this line, we asked participants to move by 10, 20, 40, 100, 1,000, 100,000, and 5,000,000 numbers, or 10, 20, and 40 minutes, 1.66 and 16 hours, 69 days, and $\approx$ 10 years on the timeline. These represent IDs of 3.3, 4.3, 5.3, 6.6, 10, 16.6, and 22.3 bits.

In addition to moving between known targets, we also first presented users with an 'initial find' task. Here, the user sees the whole timeline at once, and must navigate to an initial target. This requires no zoom out, and so assesses finding a target from minimum zoom, or, as if the user had pressed a 'reset view' or 'full zoom out' button. Strictly, this task has no fixed start point, and so no ID; the initial range of numbers seen is 20 million (40 years).

Procedure

All studies were performed on Amazon Mechanical Turk after initial lab testing. Each study was split into a mobile and a desktop version. While all mobile versions were completed on touchscreens, on the desktop version participants used whichever input device they wished. This represents real-world variability of an application-level interaction design decision. A total of 318 participants from the United States successfully completed the studies. Pay was fixed per task, which provides a soft incentive to perform the task quickly to increase wage per hour. The average pay per hour was \$8.24. Participants were allowed to participate once per study to minimize learning effects of the methods. This includes restricting participants from taking the mobile version of a study after having taken the desktop version, and vice versa. As each of the experiments had their own settings and methods, we did allow participants to take multiple studies.

Each participant was informed of the duration, pay, procedure, and goal of the study, informed of their rights, and asked for their informed consent through our institution-approved IRB protocol. For each study setting, there was a short description of the current interaction method, plus free time to play with the interaction design before the start of the experiment. Then, three sets of task 'runs' proceeded: first, a test run with an initial find and then three multi-scale pointing tasks in a random order; then, two study runs one after the other, each with an initial find and then the full set of seven multi-scale pointing task distances in a random order. These three runs allow us to observe any learning effects across the setting. On average, participants spent about 8 minutes per setting on desktop, and 13 minutes on mobile.

Study 1: Are numbers or times easier to navigate?

This study had just two settings: times (and dates), and numbers for a horizontal line. To minimize impact by other factors, we kept other settings close to a 'standard' interaction per platform: pinch to zoom and flick pan for mobile, and scroll

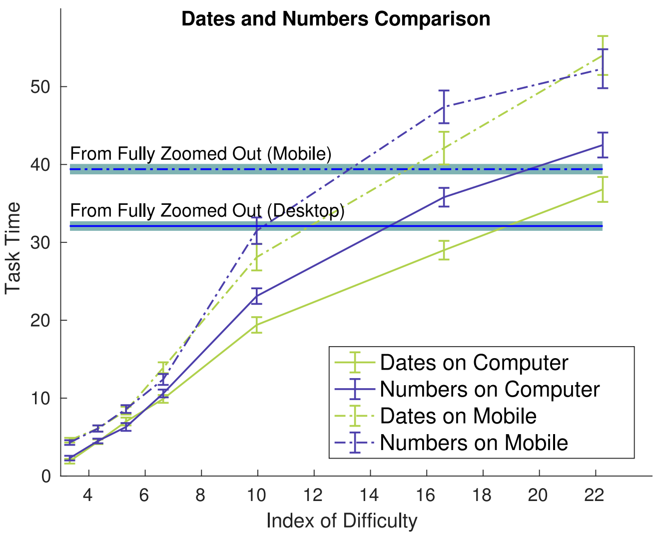


Figure 2. Number vs. dates. Error bars are standard error. The horizontal lines mark the 'initial find' time. In all cases, it takes participants longer to navigate to the farthest target than it takes for the 'initial find', showing that it would have been more efficient to navigate to the target from fully zoomed out. On both platforms, participants navigate faster using dates than numbers.

to zoom and drag pan on desktop. As such, this also provides us with a view to what a familiar interaction result looks like. 40 participants used each setting, where dates/numbers were varied within subjects. For quantitative comparison between different conditions, we averaged the task durations for each setting after removing outliers with a task duration of more than twice the average. Figure 2 shows the average task times for mobile and desktop across the different target IDs.

Task time increases approximately linearly with target ID, and tasks were consistently performed faster on computer (solid lines) than on mobile (dashed lines). For the 'initial find', there is little difference between the dates representation (blue) and numbers representation (red) on either platform. For close targets (distances 10–100), there is little difference between the dates representation (blue) and numbers representation (red) on either platform. For far targets (distances 1000–100,000), participants navigated faster using dates than numbers. On desktop, the difference is about 19.2%, and on mobile, the difference is about 11.6%. This trend seems to continue for very far targets (distance 5,000,000) on desktops, but not on mobile. Around a target distance of 10^5, the task time surpasses the task time of the 'initial find' on both platforms. This has consequences for designing high-ID interfaces, as discussed in the Design Recommendations. Based on these results, we used timelines for the remaining studies.

Th difference in performance between dates and numbers may be because it is challenging to remember long strings of numbers versus structured data like time/date [18]—people might lose track of their target numbers by mixing up one of the digits and navigate into desert fog. In our behavior analysis, Figure 5 (center bottom) shows a participant zoom in to an incorrect target, realizes their mistake, and zooming out to eventually find the correct target.

Platform	Zoom In	Zoom Out	Pan
	Pinch		Flick
	Wheel		Linear
	Hold	Click + Hold	Standard
	Dynamic Zoom		Standard
	Brush	2x Click	Pinch
	Brush	2x Click	Wheel
	2x Click	Hold	Standard
	Rubbing X	Rubbing Y	Standard

Table 2. Interaction combinations used in study three. 'Standard' pan refers to the typical device defaults of flick pan on mobile, and drag pan on desktop.

Study 2:

Is timeline pan/zoom easier horizontally or vertically?

We compare timelines with different orientations, again with devices taking their 'standard' interactions. However, we also pick an additional interaction combination for each platform where we assume performance to be affected by the orientation change. For mobile, we chose dynamic zoom (zoom control orthogonal to timeline) and flick pan. For desktop, we chose brush zoom and scroll pan. These choices are not intended to be equivalent across devices; merely illustrative of a change in interaction performance. 40 participants used each setting, where interaction was varied within subjects. As in study 1, for quantitative comparison between different conditions, we averaged the task durations for each setting after removing outliers with a task duration of more than twice the average.

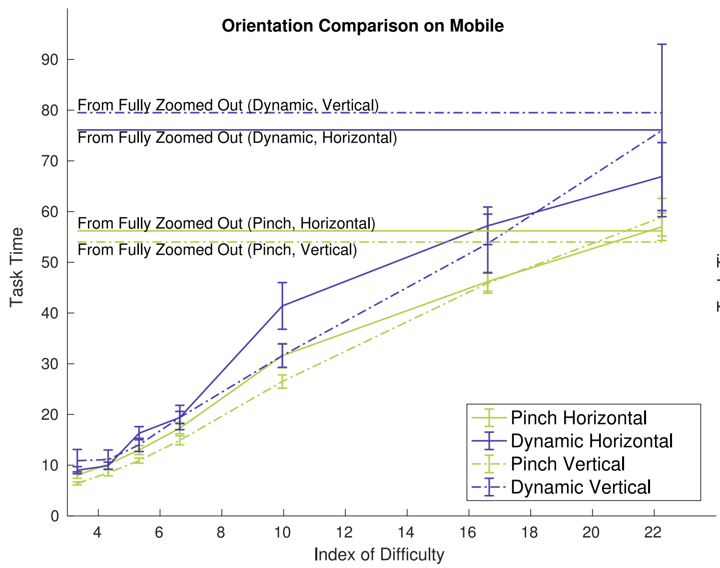
For mobile (Figure 3, left), the difference between vertical (dashed) and horizontal (solid) orientations is small for the familiar pinch zoom (lime), but is larger and with higher performance variability for the less familiar and orthogonal dynamic zoom (navy), as expected. Across both interactions, vertical orientation performs better on average. For desktop (Figure 3, right), again we found that the more significant impact was interaction method (lime vs. navy), rather than orientation (solid vs. dashed). While we did not find a large difference for the performances using scroll zoom (lime), the impact on brush zoom is greater. participants using the vertical orientation navigating to their targets faster.

Since the orientation mattered less to overall performance, but that the vertical orientation worked faster for the orientation-dependent method of dynamic zoom on mobile, our main study was performed in the vertical orientation.

Study 3:

How effective are different pan and zoom techniques?

To begin this study, we had to pick a subset of combinations to test, as Table 1 suggests there are simply too many to compare in a reasonable amount of participant time. We aimed to evaluate each major zoom in method on both platforms—*hold zoom, dynamic zoom, brush zoom, double click zoom*, and a gesture as *rub zoom*. Given that as a constraint, paired each with the most complementary zoom out and pan methods (as assessed by our judgment and in toy lab trials). Table 2 lists our final combinations.



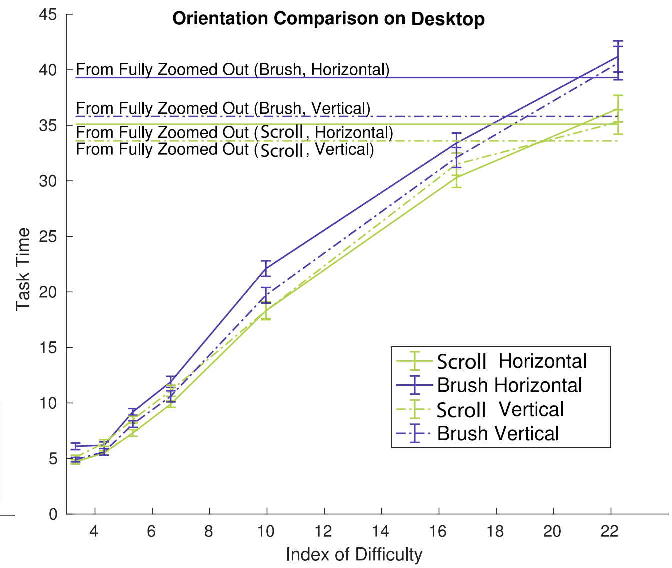


Figure 3. Comparison of task times using two techniques and two different visualization orientations on mobile (left) and computer (right). On both platforms, the effect of the interaction method (lime versus blue) dominates over the effect of the orientation (solid versus dashed line). On both platforms, the standard method (lime) is relatively unaffected by the orientation. On both platforms, the alternative technique we expected to be more affected by the orientation (navy) did in fact showed greater variation for the two orientations. In both cases, participants performed the tasks faster using the vertical orientation.

While we wished to maintain a within-subjects design, six combinations per platform would lead to a total study time per participant of 80 minutes on mobile. At this length, fatigue effects are likely to affect any differences between interaction designs. As such, we doubled the participant pool and split the study into two sets. Each set compared three interaction techniques to the 'standard' platform method. We present our results in aggregate. 40 participants performed each platform and each set on desktop, with 39 on mobile, for a total of 158 participants.

For analysis, we computed the difference between a participant's task time using the standard setting (scroll zoom on computer, pinch zoom on mobile) and their other interaction designs. This is to counteract learning effects that users may have had from participating in one of the earlier studies, and minimizes the error due to the correlations in each participant's data. It also removes one line from a complex visualization task. Again, we removed outliers with a task duration of more than twice the average. We present mean average performance along with variability as standard error (Figure 4).

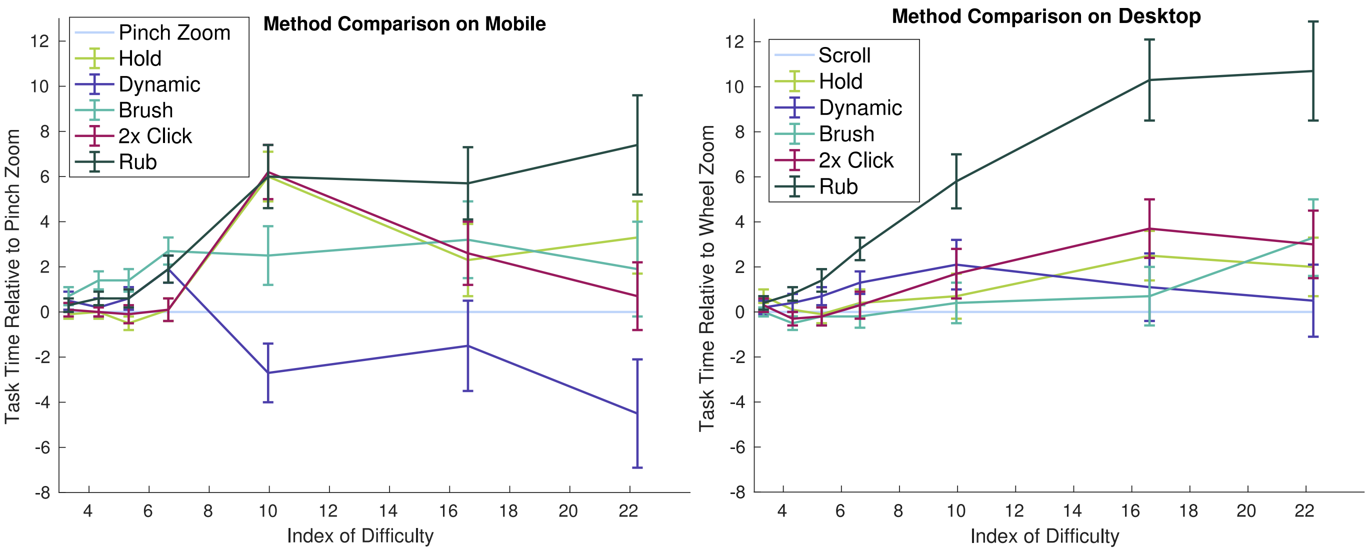
On mobile, for the initial find, double click zoom is approximately equivalent to pinch zoom, with all other techniques being slower (see table). All methods perform reasonably well for short distance targets. For far targets with IDs 10–22, dynamic zoom dominates and is about 3 seconds faster than any other method, even familiar pinch zoom. Pinch zoom is second fastest, with hold, dynamic, and double click zooms being comparable for third place. On desktop, for the initial find, dynamic zoom is approximately equivalent to scroll zoom. Again, for low-bit IDs, most techniques are comparable, but only dynamic zoom remains comparable to scroll zoom at high IDs.

Learning Effects

We measured learning effects by looking at how the performance of each interaction design varied against the average interaction design performance, over the time that the participant had spent with each interaction design. First, averaged across all designs and participants, performance increased by over two seconds over the 10 minutes of each trial. Furthermore, our analysis seems to indicate that participant performance variability across interaction methods decreased over time, which suggests that interaction methods would benefit from further training. Across studies, we found that participants that had previously participated in either study 1 or study 2 on average performed 7 seconds faster in study 3 using the standard method of pinch zoom on mobile.

Participant Feedback

Of 81 comments, 23 comments included the word "fun". Participants generally found it interesting to use unusual methods of zooming: "Interesting study! New methods of zooming in and out were fun to use and seemed just as good as the standard way". Many had more specific preferences, although there was no general consensus: brush zoom "was more accurate since I could point at the exact spot" versus "really, *really,* did not like the brushing/double-click method—felt excruciatingly counterintuitive". Or, "I thought using the scroll wheel was the easiest to use because it was the most simple method" versus "it was exhausting waiting for the zoom in and out on a few of the methods. the worst was the scrollwheel". The desire to control the speed of the zoom interaction was brought up 3 times. Another complaint was the physical work required to use pinch zoom on mobile: "Two finger pinch mode is way too cumbersome", and "oh man, my screen is going to be smeary now". We are not confident that future research will consider screen buffing as an interaction cost.



Task (ID)	Pinch	Hold	Dynamic	Brush	2x Click	Rub
Initial Find	0	2.3 ± 1.4	2 ± 2.2	2.4 ± 1.7	-0.4 ± 1.3	4.5 ± 2.2
3.32	0	-0.1 ± 0.2	0.5 ± 0.4	0.7 ± 0.4	0.1 ± 0.3	0.3 ± 0.3
4.32	0	0 ± 0.3	0.2 ± 0.4	1.4 ± 0.4	0 ± 0.2	0.6 ± 0.3
5.32	0	-0.5 ± 0.3	0.6 ± 0.5	1.4 ± 0.5	-0.1 ± 0.4	0.6 ± 0.4
6.64	0	0.1 ± 0.5	1.9 ± 0.6	2.7 ± 0.6	0.1 ± 0.5	1.9 ± 0.6
9.96	0	6 ± 1.1	-2.7 ± 1.3	2.5 ± 1.3	6.2 ± 1.2	6 ± 1.4
16.61	0	2.3 ± 1.6	-1.5 ± 2	3.2 ± 1.7	2.6 ± 1.4	5.7 ± 1.6
22.25	0	3.3 ± 1.6	-4.5 ± 2.4	1.9 ± 2.1	0.7 ± 1.5	7.4 ± 2.2

Scroll	Hold	Dynamic	Brush	2x Click	Rub
0	4.1 ± 1.6	-0.2 ± 1.4	3.6 ± 1.5	3.1 ± 1.2	9.1 ± 2.1
0	0.7 ± 0.3	0.2 ± 0.3	0 ± 0.2	0.3 ± 0.3	0.4 ± 0.3
0	0.1 ± 0.3	0.4 ± 0.4	-0.5 ± 0.3	-0.3 ± 0.3	0.8 ± 0.3
0	-0.1 ± 0.4	0.7 ± 0.4	-0.2 ± 0.4	-0.2 ± 0.4	1.4 ± 0.5
0	0.4 ± 0.6	1.3 ± 0.5	-0.2 ± 0.5	0.3 ± 0.6	2.8 ± 0.5
0	0.7 ± 1	2.1 ± 1.1	0.4 ± 0.9	1.7 ± 1.1	5.8 ± 1.2
0	2.5 ± 1.1	1.1 ± 1.5	0.7 ± 1.3	3.7 ± 1.3	10.3 ± 1.8
0	2 ± 1.3	0.5 ± 1.6	3.3 ± 1.7	3 ± 1.5	10.7 ± 2.2

Figure 4. Left: mobile, right: computer. Top: visual comparison. Bottom: tables that also include the initial find comparison. Tables: Cells are green if the average task time of that method is within 2 seconds of the fastest method for that task, within 4 seconds for orange, and red for anything slower than that. Errors are shown using a single standard derivation, or $Z = 1$.

Behavior Analysis

To better understand these results and analyze participant behavior, we show visualizations of a participant's pan, zoom in, and zoom out actions, plus their idle time, in the multi-scale point tasks (Figure 5 shows three different behaviors). Further, we show *all* participant behaviors in a study scenario (Figure 6), where we can identify different behavior patterns across interaction types by observing color distributions among participants. Please view these figures digitally and zoom in. Our website contains plots for each participant trial, along with the ability to replay the interaction on the timeline. Finally, we show pan, zoom in, zoom out, and idle time aggregated over each condition as a percentage of total interaction time (Figure 7). Here, it is simple to see that zooming was rarely used below our target ID of 6.6 bits, which is a little lower than the 8 bits found by Guiard et al. [7].

Figure 6 allows us to derive characteristics for each interaction method. We draw attention to three main attributes: First, the **angle** of the distribution shows the variability in performance among a real world population. Second, the **speed of the fastest performers** is visible at the top of each interaction, provides us an indication of relative peak performance of a technique. Third, the amount of **time spent idle** (light blue) helps us see potential for improvement, either through training or through better visualization to minimize cognitive load.

Angle: The greater the variation among participants' task times, the shallower the angle of the right edge of each plot. Easier to use interactions have a steeper angle because they induce more even performance across participants. Looking at these angles reveals that pinch, double click, and hold zoom are easier to learn, while rub, brush, and dynamic zoom are harder to learn.

Speed of fastest performers: Pinch, double click, and hold have relatively worse peak performance, and rub, brush, and dynamic zoom all have participants that perform faster than those of other methods. This suggests that they may be more expert interactions, and that better training would improve the overall population performance.

Time spent idle Pinch, scroll, and hold zoom timelines are packed very densely with manipulation interactions. Brush zoom is packed very sparsely, which suggests that the interaction/visualization may be optimized. Rub zoom is very densely packed as it is a cyclic gesture, though not all of this time. Dynamic zoom and rub zoom have red/green speckling, which is a sign that some users struggled to zoom in to the correct location.

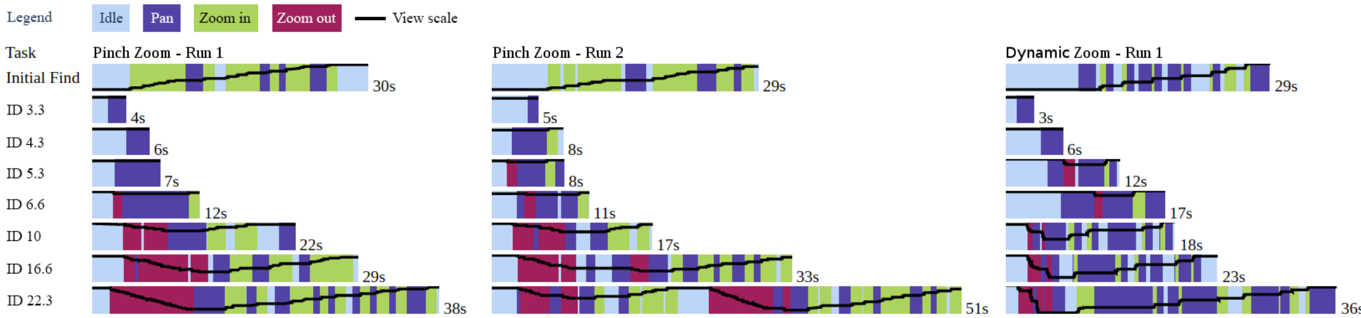


Figure 5. Three example timelines showing navigation behaviour for one participants. Each row represents one task distance, or a target with a specific distance from the current position. The black line denotes the scale of the view over time: When the line is at the top, the target is large on the screen; at the bottom, all 20.000.000 numbers (40 years) are visible. *Left:* In "Pinch Zoom - Run 1", the participant shows typical behaviour. *Middle:* In "Pinch Zoom - Run 2" ID 22.3, a 'desert fog' issue is visible. After zooming out by about 30%, the participant thinks the target is within view and zooms back in all the way to select the target. After short inactivity, the participant realized that they navigated to the wrong number and starts zooming out. In the first attempt, the participant did not zoom out far enough, whereas this time the participant zooms out much farther and finally succeeds in zooming in to the target. *Right:* Using a different interaction technique in "Dynamic Zoom - Run 1", different behaviour is visible with faster zooming and longer panning periods.

Initial find: An optimal interaction would not include any zooming out (red), and we can see that specially in rub and dynamic zooming, it occurs on many participants' timelines. Again, this may indicate interaction difficulty. For dynamic zoom, we believe that participants may have zoomed in too quickly, and users had to zoom back out to find their path. For rub zoom, the implementation used here allows for very easy switching from zooming in to zooming out, and some participants may have switched accidentally.

DESIGN IMPLICATIONS

Our findings have implications for designers of application that require panning and zooming. From the number vs. timelines study results in Figure 2, we have two recommendations: One, long timelines need some visual aid for navigation. Whenever possible, designers should display information in chunks that can be individually processed, as opposed to long numbers. In this case, switching to dates improved performance.

Our second recommendation is based on the observation that the task time for high-ID targets surpasses the task time of the 'initial find' on both platforms beyond ID = 18. For targets with IDs greater than 18, it would be more efficient for participants to reset the zoom and then zoom in again, rather than to partially zoom out from their location. One might assume that participants would be aware of this fact; however, after investigating the data, we found that participants in this study only zoomed out as far as they needed to. Application developers should use this knowledge to determine if an option to zoom out fully may be advantageous for their application, and possibly add either a manual or even an automatic way of resetting the view for some cases. Possible future research could reveal whether manual zoom out buttons are used effectively.

Drawing on our results in Figure 3, we recommend using vertical timelines when possible, specially for advanced techniques such as dynamic zoom and brush zoom. For the standard methods, the difference is small.

Based on the results of study 3 in Figure 4 and Figure 8, we recommend that application developers use "A"-graded interaction methods for their tasks. The results shown in Figure 4 mostly apply to novice users, as our participants only interacted with the timeline for under 15 minutes. However, the distributions in Figure 6 allow us some indication of the relative peak performances to give recommendations for applications with expert users.

On mobile, we recommend to use pinch and double click zoom to optimize task time for novice users for initial and close targets. Unfortunately, both techniques typically have an upper limit in how much scale change they can cause with a single interaction. For an application that scales to expert users, or in applications in which users need to switch between targets that are far from each other, we recommend the use of dynamic zoom.

For applications in which it is important to use familiar techniques such as pinch zoom, care must be taken to use this interaction method in long timelines with extensive use. Because a high amount of timeline manipulation is needed using pinch and hold zoom, long time users are limited in how much they can improve their interactions. To make these interactions more scalable, this core problem needs to be addressed, and options such as a non-linear mapping between finger motion and timeline scale change need to be considered.

On desktop, we recommend scroll zoom and dynamic zoom for all tasks. Dynamic zoom requires familiarity, and so applications with many short-time users may prefer a standard scroll zoom. However, applications with users that stay for at least 15 minutes and ideally longer should consider using dynamic zoom to enable users with experience to navigate faster.

The two methods marked "A" for "Expert"—dynamic and brush zoom—both enable the user to make vast scale changes with a single interaction. This enables fast navigation, but can also have drawbacks: Fast context switches can create the need for users to re-orientate themselves, thus increasing the time without manipulating the timeline. We believe this to be visible in the timeline for brushing in Figure 6, where

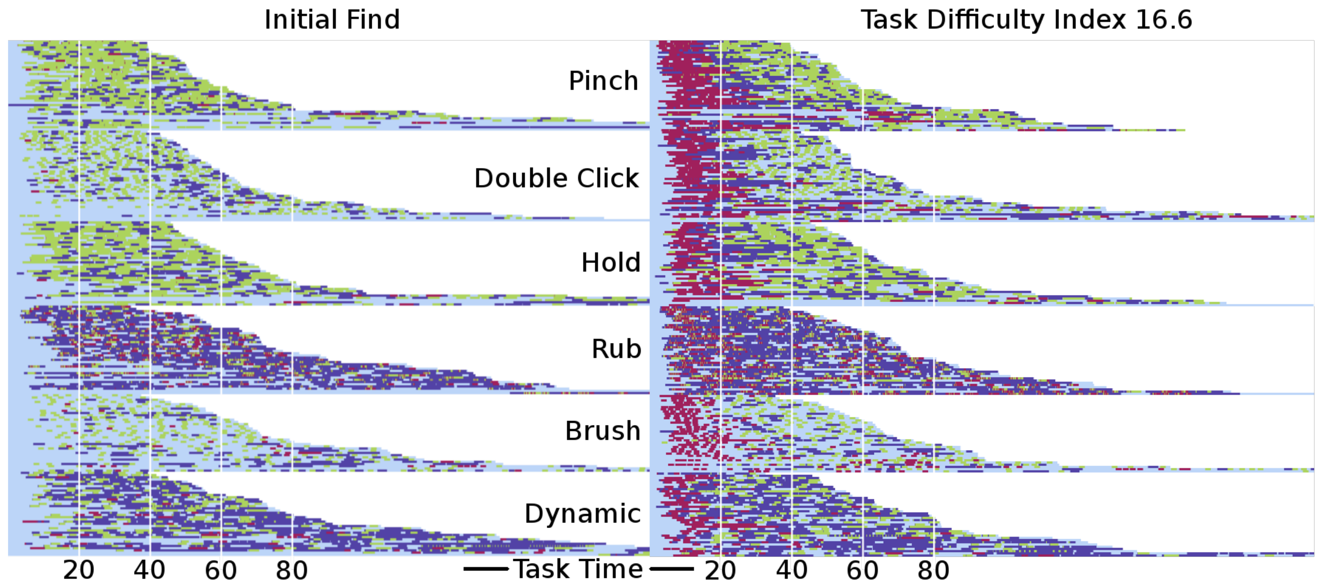


Figure 6. Participant behavior for each of the 39 mobile participants in study 3, sorted by task time. Colors are as in Figure 5. We show a single high ID for each interaction type. Each participant's behavior is visible on zoom; however, the goal of this plot is to see trends across participants by looking broadly at the distributions of colors. For instance, we can see that brush zoom requires significantly less zoom out (less red) *and* less zoom in (less green), but that it requires more idle time to process the zoom operations (more blue).

most of the timeline contains to manipulations. This is even though our implementation uses an animation to make the context switch less dramatic, albeit a fast one at 400ms. We recommend taking care when designing context switches, and using a longer animation time.

To further unlock the potential of techniques that require little interaction time, designers must aid users in visual navigation as much as possible, e.g., by providing visual cues to anchor the zoom operation, or by animating at a speed which reduces the need for data reorientation.

DISCUSSION AND LIMITATIONS

There is an approximately logarithimic relationship between the the distance traveled to select the next target, and the amount of time it takes to find that target. The relationship is observable in Figures 2 and 3, where the plot line is close to linear in a log-linear scale plot. This is compatible with the well-known Hick's Law [9] that the decision time to choose between a number of choices (e.g., in a menu) is logarithmically proportional to the number of choices. The key difference is that substantial time is required to navigate to the number or date to be selected, which overshadows the decision time. Selection between a large range of numbers (as in these studies) is likely to take time linearly proportional to the size of the range if panning was the only interaction offered, but having a zoom interaction returns the time needed back into a logarithmic relationship again.

Participant performance depends on parameters, and also on the specific implementation. It is likely that dedicated tweaking could find parameters which would make a method better than in this evaluation. Through limited in-person pilot studies, we aimed to choose the best settings for each interaction method. We release our library in part to hope to reduce this implementation/parameter problem. Participant performance can also depend on device compute speed. We measured device performance in JavaScript for each participant, but participant performance was not significantly correlated with compute device speed.

CONCLUSION

We have investigated interaction techniques for navigating a large range of values on desktop and mobile. These pan and zoom techniques can be used in interfaces where precisely and quickly selecting a specific date or number within a large space is important, such as timeline sliders. As part of the work, we release an open source library for experimentation, and for incorporating 12 interaction techniques. We describe the differences in effectiveness between selecting dates and numbers, between horizontal and vertical timelines, and the interaction techniques. This leads to a set of guidelines about which technique to use depending on the numerical distance of the target, and the form factor of the computer.

Future studies can build on this work, and take it in several directions. We recommend three directions of research. One interesting question is whether existing techniques that users are familiar with can be expanded upon to improve their performance for expert users without giving up the benefit of a familiar technique. To reduce the need for a lot of interaction in methods like pinch zoom, the linear mapping between finger motion and scaling could be replaced with a quadratic or higher-dimensional function. This technique is already implemented in the EASYPZ library and is ready to be compared against other methods.

A second important task is to investigate the techniques from which expert can benefit. A longer-term study is needed to

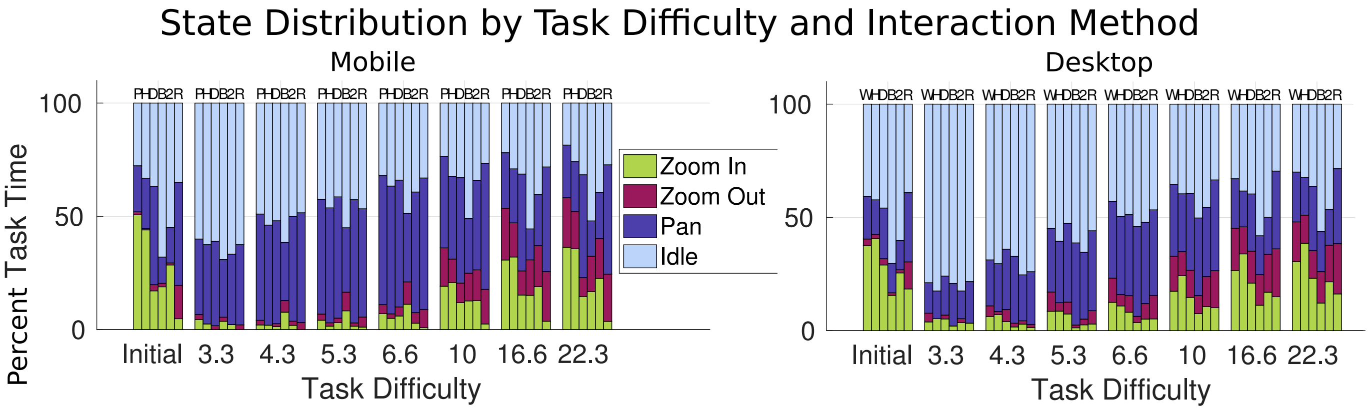


Figure 7. Aggregated subtask interactions across mobile and desktop, for each ID, and for each interaction type, as a percentage of total task time. Here, we can see clearly that brush zoom (B) requires significantly more participant idle time. We attribute this to the rapid change in data view that brush zoom induces, which causes participant to need to reorient. Mobile: P = Pinch Zoom. H = Hold Zoom. D = Dynamic Zoom. B = Brush Zoom. 2 = Double-click Zoom. R = Rub Zoom. Desktop: W = Scroll Zoom.

Method				Mobile			Desktop			Audience		
	Zoom In	Zoom Out	Pan	Initial	Near	Far	Initial	Near	Far	Beginner	Intermediate	Expert
	Pinch		Flick	A	A	B	-	-	-	A	B	C
	Wheel		Linear	-	-	-	A	A	A	A	B	C
	2x Click	Hold	Standard	A	A	C	B	A	B	A	B	B
	Hold	Click + Hold	Standard	B	A	C	C	A	B	B*	B	C
	Dynamic		Standard	B	A	A	A	A	A	B	A	A
	Brush	2x Click	Pinch	B	B	C	-	-	-	B+	A	A
	Brush	2x Click	Wheel	-	-	-	B	A	A	B+	A	A
	Rubing X	Rubbing Y	Standard	C	A	C	C	B	C	C	A	B

Figure 8. Grades of how appropriate each interaction technique would be for the different form factors, various travel distances, and the skill level of the users. "A" grade scenarios are preferred for practical applications, while "C" grade techniques are discouraged from use. Grades for the initial, near and far targets are averaged from results in Figure 4, with "Near" denoting index of difficulty 3-7, and "Far" denoting 7-20 (excluding results slower than the initial find). Grades for different audiences are estimates based on results in Figure 6. "Beginner" denotes a good average speed for first time users. "Intermediate" denotes good medium-term returns on learning the technique. "Expert" denotes great potential time savings when the interaction flow is optimized. B*: We believe that the choice of click + hold to zoom out may have confused some users in combination with hold to zoom in. A different zoom out input may have allowed hold to zoom in to be rated "A". ***Note:*** **'Standard' pan refers to the typical device defaults of flick pan on mobile, and drag pan on desktop.**

be able to give definitive answers to the question of highest-performing methods and settings.

Our third and final recommendation is to try to bridge the gap between expert and beginner techniques. One aspect of this is to look into possible combinations of methods, e.g. by allowing dynamic zoom at the same time as pinch zoom to target novice and expert users, and investigating the usage of each. Another way to bridge this gap is with adapted and new techniques that naturally behave like well-established techniques, but which can also be used more non-linearly.

As we accumulate more data in the form of passively-recorded videos, growing sets of photos, and digital documents and records that keep expanding, we will need ways to efficiently navigate to the moment or position we want. This work takes a step towards finding the right interactions for the right time.

REFERENCES

1. Christopher Ahlberg and Ben Shneiderman. 1994. The Alphaslider: A Compact and Rapid Selector. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '94)*. ACM, New York, NY, USA, 365–371. DOI: http://dx.doi.org/10.1145/191666.191790

2. Caroline Appert and Jean-Daniel Fekete. 2006. OrthoZoom Scroller: 1D Multi-scale Navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. ACM, New York, NY, USA, 21–30. DOI: http://dx.doi.org/10.1145/1124772.1124776

3. Benjamin B. Bederson and James D. Hollan. 1994. Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics. In *Proceedings of the 7th Annual ACM Symposium on User Interface Software and*

Technology (UIST '94). ACM, New York, NY, USA, 17–26. DOI:http://dx.doi.org/10.1145/192426.192435

4. Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. 2009. A Review of Overview+Detail, Zooming, and Focus+Context Interfaces. *ACM Comput. Surv.* 41, 1, Article 2 (Jan. 2009), 31 pages. DOI: http://dx.doi.org/10.1145/1456650.1456652
5. George W. Furnas and Xiaolong Zhang. 1998. MuSE: A Multiscale Editor. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology (UIST '98)*. ACM, New York, NY, USA, 107–116. DOI:http://dx.doi.org/10.1145/288392.288579
6. Google. 2015. YouTube. https://www.youtube.com/watch?v=mpbWQbkl8_g#t=20m15s. (2015). Accessed: 14th September 2017.
7. Yves Guiard and Michel Beaudouin-Lafon. 2004. Target acquisition in multiscale electronic worlds. *International Journal of Human-Computer Studies* 61, 6 (2004), 875 – 905. DOI:http://dx.doi.org/https://doi.org/10.1016/j.ijhcs.2004.09.005 Fitts' law 50 years later: applications and contributions from human-computer interaction.
8. Chris Harrison and Anind K. Dey. 2008. Lean and Zoom: Proximity-aware User Interface and Content Magnification. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 507–510. DOI: http://dx.doi.org/10.1145/1357054.1357135
9. William E Hick. 1952. On the rate of gain of information. *Quarterly Journal of Experimental Psychology* 4, 1 (1952), 11–26.
10. Ken Hinckley and Hyunyoung Song. 2011. Sensor Synaesthesia: Touch in Motion, and Motion in Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 801–810. DOI: http://dx.doi.org/10.1145/1978942.1979059
11. Wolfgang Hürst. 2006. Interactive Audio-visual Video Browsing. In *Proceedings of the 14th ACM International Conference on Multimedia (MM '06)*. ACM, New York, NY, USA, 675–678. DOI: http://dx.doi.org/10.1145/1180639.1180781
12. Takeo Igarashi and Ken Hinckley. 2000. Speed-dependent Automatic Zooming for Browsing Large Documents. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST '00)*. ACM, New York, NY, USA, 139–148. DOI:http://dx.doi.org/10.1145/354401.354435
13. Neel Joshi, Abhishek Kar, and Michael Cohen. 2012. Looking at You: Fused Gyro and Face Tracking for Viewing Large Imagery on Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2211–2220. DOI: http://dx.doi.org/10.1145/2207676.2208375
14. Susanne Jul and George W. Furnas. 1998. Critical Zones in Desert Fog: Aids to Multiscale Navigation. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology (UIST '98)*. ACM, New York, NY, USA, 97–106. DOI: http://dx.doi.org/10.1145/288392.288578
15. Myron W. Krueger, Thomas Gionfriddo, and Katrin Hinrichsen. 1985. VIDEOPLACE&Mdash;an Artificial Reality. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '85)*. ACM, New York, NY, USA, 35–40. DOI: http://dx.doi.org/10.1145/317456.317463
16. Sylvain Malacria, Eric Lecolinet, and Yves Guiard. 2010. Clutch-free Panning and Integrated Pan-zoom Control on Touch-sensitive Surfaces: The Cyclostar Approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 2615–2624. DOI: http://dx.doi.org/10.1145/1753326.1753724
17. Toshiyuki Masui, Kouichi Kashiwagi, and George R. Borden, IV. 1995. Elastic Graphical Interfaces to Precise Data Manipulation. In *Conference Companion on Human Factors in Computing Systems (CHI '95)*. ACM, New York, NY, USA, 143–144. DOI: http://dx.doi.org/10.1145/223355.223471
18. George A Miller. 1956. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review* 63, 2 (1956), 81.
19. Mathieu Nancel, Julie Wagner, Emmanuel Pietriga, Olivier Chapuis, and Wendy Mackay. 2011. Mid-air Pan-and-zoom on Wall-sized Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 177–186. DOI:http://dx.doi.org/10.1145/1978942.1978969
20. Alex Olwal, Steven Feiner, and Susanna Heyman. 2008. Rubbing and Tapping for Precise and Rapid Selection on Touch-screen Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 295–304. DOI: http://dx.doi.org/10.1145/1357054.1357105
21. Paul Over, Jonathan Fiscus, David Joy, Martial Michel, George Awad, Wessel Kraaij, Alan F. Smeaton, Georges Quénot, and Roeland Ordelman. 2015. TRECVID 2015 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics. In *Proceedings of TRECVID 2015*. NIST, USA.
22. Ken Perlin and David Fox. 1993. Pad: An Alternative Approach to the Computer Interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93)*. ACM, New York, NY, USA, 57–64. DOI: http://dx.doi.org/10.1145/166117.166125

23. Philip Quinn, Sylvain Malacria, and Andy Cockburn. 2013. Touch Scrolling Transfer Functions. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 61–70. DOI: http://dx.doi.org/10.1145/2501988.2501995

24. Gonzalo Ramos and Ravin Balakrishnan. 2003. Fluid Interaction Techniques for the Control and Annotation of Digital Video. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology (UIST '03)*. ACM, New York, NY, USA, 105–114. DOI: http://dx.doi.org/10.1145/964696.964708

25. Heather A. Richter, Jason A. Brotherton, Gregory D. Abowd, and Khai N. Truong. 1999. *A Multi-Scale Timeline Slider for Stream Visualization and Control*. Technical Report. Georgia Tech GVU Center.

26. Klaus Schoeffmann, Marco A. Hudelist, and Jochen Huber. 2015. Video Interaction Tools: A Survey of Recent Work. *ACM Comput. Surv.* 48, 1, Article 14 (Sept. 2015), 34 pages. DOI:http://dx.doi.org/10.1145/2808796

27. Martin Spindler, Martin Schuessler, Marcel Martsch, and Raimund Dachselt. 2014. Pinch-drag-flick vs. Spatial Input: Rethinking Zoom & Pan on Mobile Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1113–1122. DOI: http://dx.doi.org/10.1145/2556288.2557028