Introducing Transient Gestures to Improve Pan and Zoom on Touch Surfaces

Jeff Avery, Sylvain Malacria, Mathieu Nancel, Géry Casiez, Edward Lank

Introducing Transient Gestures to Improve Pan and Zoom on Touch Surfaces

Jeff Avery[1], **Sylvain Malacria**[2], **Mathieu Nancel**[2], **Géry Casiez**[3], **Edward Lank**[1]
[1]University of Waterloo, Canada, [2]Inria, France, [3]Université de Lille, France
{jeffery.avery, lank}@uwaterloo.ca, {sylvain.malacria, mathieu.nancel}@inria.fr,
gery.casiez@univ-lille.fr

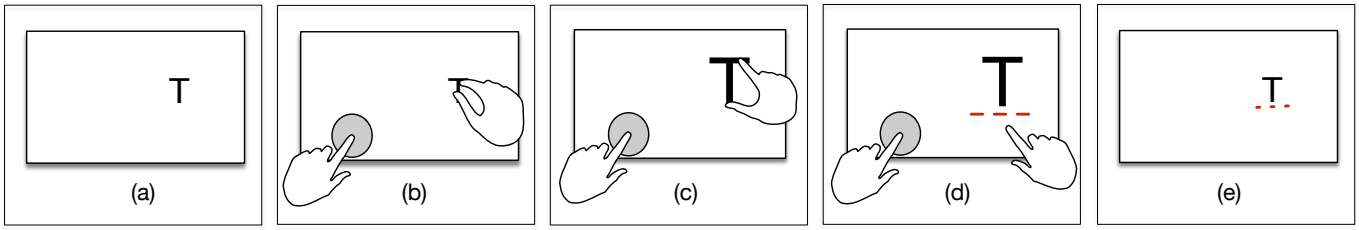

Figure 1. Illustration of transient zooming. The user wants to precisely annotate the letter T (a); she bookmarks the starting viewport using her non-dominant hand (b), zooms in, (c) annotates the T (d), and then lifts her fingers to restore the bookmarked viewport (e). Normal operations can be performed before, during or after the bookmark operation.

ABSTRACT
Despite the ubiquity of touch-based input and the availability of increasingly computationally powerful touchscreen devices, there has been comparatively little work on enhancing basic canonical gestures such as swipe-to-pan and pinch-to-zoom. In this paper, we introduce transient pan and zoom, i.e. pan and zoom manipulation gestures that temporarily alter the view and can be rapidly undone. Leveraging typical touchscreen support for additional contact points, we design our transient gestures such that they co-exist with traditional pan and zoom interaction. We show that our transient pan-and-zoom reduces repetition in multi-level navigation and facilitates rapid movement between document states. We conclude with a discussion of user feedback, and directions for future research.

ACM Classification Keywords
H.5.2. Information Interfaces and Presentation: User Interfaces; Interaction styles

Author Keywords
tablet; multitouch; bimanual input

INTRODUCTION
Modern devices almost systematically rely on multi-touch input vocabulary. However, despite their success, there has been relatively little work on optimizing basic canonical gestures. Multi-touch gesture sets are often limited to simple gestures like tap, double-tap, drag and pinch/spread. These gestures are easy for novices to learn, but may not be sufficient for more demanding or repetitive tasks, particularly by expert users.

Pan-and-zoom navigation is an interesting example. It is the standard gesture for multi-scale navigation, used by both novice and expert users across use cases, such as map navigation, drawing tasks, and document annotation. One common optimization is double-tap or contextual zoom, to snap between two fixed zoom levels. While this is simple to understand and easy to adopt, it requires target-aware systems [18] that are designed around assumptions of target zoom level. This may work for very specific scenarios (e.g. map navigation), but may not be flexible enough for expert usage, where the user requires precise control over the target zoom level. Other research has been done to optimize pan-and-zoom [3, 9, 22], but few of these optimizations have become mainstream.

Our work expands the design space around canonical gestures such as pan and pinch-to-zoom, to add functionality for expert use. Quickly changing zoom level, or shifting between areas of interest is a common strategy in information-rich displays used by expert users (e.g. real-time strategy games, image editing applications). However, these actions often suffer from inefficiencies. Previous work has demonstrated that users need to clutch excessively when performing data manipulations [3, 16]. For example, suppose a pinch is done and content is lost on the screen, or the user zooms in to some content and then wishes to re-perform the zoom to look at additional low-level details. In both cases, excessive zoom and pan operations would be needed to reposition the view and allow the user to find the correct data.

We explore the use of expert gestures that support fluid transitions between document states and eliminate the need for excessive clutching (Figure 1). We focus on tablet interaction, where this type of complex interaction is commonly used.

Our transient gestures allow users to quickly undo the effects of pan and zoom operations, and support manipulations that can easily be undone during interaction without recourse to clutching. This provides support for more complex scenarios, like real-time strategy gaming (where the user needs to quickly reposition the viewport around content at different locations and resolutions), image editing, and active reading (where the user needs to navigate around a document efficiently) [19]. Our hypothesis is that a transient zoom will decrease effort (e.g. number of actions) and time required to perform these types of tasks, compared to standard pan-and-zoom techniques.

The paper is organized as follows. After reviewing related work, we present the design of our transient gestures, including an initial pilot and a revised design based on user feedback. We then report on an experiment where we compare standard pan-and-zoom, double-tap to zoom, and our transient pan-and-zoom, and demonstrate the performance of transient techniques over traditional navigation techniques. We present the results of a post-experiment survey, including user feedback on these techniques. Finally, we conclude with a discussion of future directions for this research.

Our contributions in this paper consist of:

- An exploration and iteration of transient gesture designs.
- The design of bimanual gestures that support fluid movement between transient and atomic operations.
- The description and realization of a system supporting expert gestures.
- An evaluation of performance and user satisfaction with these gestures.

RELATED WORK

Panning and pinch-to-zoom has emerged as the standard way of directly manipulating and transforming content on multi-touch devices. Although occlusion issues are well-known [21], these gestures also present problems when dealing with focus+context tasks [12]. Users can use pinch-to-zoom to zoom in to view details for a task, and pan while interacting with content, but often need to zoom back out again to an overview level before proceeding with a task [14, 10]. This need to snap in and out and clutch repeatedly is inefficient and represents wasted effort [3, 9, 22].

Previous work has attempted to address this problem by reducing the number of actions required to complete a task. Pinch-to-Zoom Plus performs automatic document manipulations (zoom acceleration and automatic panning) to reduce the number of actions [3]. Similarly, FingerGlass [10] and CycloStar [14] provide more efficient replacements for standard zoom functionality. ZoomPointing [1] uses a bounding box to delineate a zoom region, so that panning isn't required.

Although effective at reducing the effort required, these techniques all attempt to replace standard gestures. We believe that pinch-to-zoom and pan have the advantage of being familiar and well-understood, and any attempt to replace them outright will meet resistance. Expert users have shown a willingness to adopt new gestures [4], but novice users are less likely to do so. Our gestures are intended to augment the existing gesture set with advanced gestures that expert users may be willing to adopt, and which are also designed to aid in the transition from novice to expert use [6].

The idea of a transient technique is not new. Offset [17] is a pinch-to-zoom replacement that attempts to pan content while zooming, and automatically restores the initial resolution and position after the user lifts their hands. However, this technique does not allow clutching, which limits its applicability, and, again, it replaces standard pinch-to-zoom behavior. Appert et al. identify the value in restoring state for drag-drop operations, but do not address touch interactions like pan and zoom [2].

FingerGlass [10] allows the user to use his non-dominant hand to define a zoom region, his dominant hand for interaction, and to snap-back by releasing both hands. Our transient mode is loosely based on a similar style of interaction that keeps the dominant hand free to perform a task. However, our technique allows for successive and cumulative pan and zoom operations, which may be required for complex tasks.

Our initial design for transient pan-and-zoom was motivated by the bezel pins and swipes introduced in BiPad [23], and our prototype used similar gestures to control the transition between states. We also leveraged previous work when deciding to define a bimanual interaction style where the off-hand anchors and defines the task for the dominant hand [7, 8, 20, 10].

DESIGNING TRANSIENT PAN-AND-ZOOM

Transient pan and zoom is designed around a bookmarking metaphor, where we allow users to define a pre-specified UI state as a bookmark to which they can return. In this, it is similar to UIMarks [5], a mouse-and-keyboard interaction technique where interface elements (e.g. toolbar buttons) are specified as a target where the pointer can be teleported.

We conducted a series of pilot studies to motivate the design of our transient pan and zoom techniques. From these pilot studies, we identified the following design principles:

- Avoid automatic mode switches and allow users to explicitly control when they enter and leave transient mode.
- Use the off-hand for triggering transient mode and leave the dominant-hand free to perform normal operations.
- Respect and mirror the semantic meaning of gestures in our transient gesture (e.g. transient pan should resemble standard pan) to facilitate a transition to expert-level behaviour through rehearsal of familiar, novice gestures [11].
- Implement transient as a quasi-mode that can be layered over existing gestures.

Figure 1 demonstrates the final transient technique, which complements standard pan-and-zoom functionality. One-finger is

reserved for interacting with the view (e.g. dragging, activating controls), while two-fingers are used for panning, zooming and other standard operations.[1]

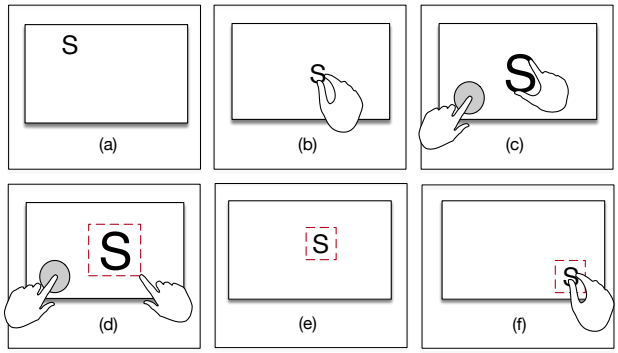

Figure 2. Transient pan/zoom design supports interleaving atomic and transient actions. From a starting screen (a), a user performs a two-finger pan to center content (b); she then holds down third finger to enable transient mode, indicated by a shaded region under the pinning finger, and performs a transient zoom (c) before drawing (d). Lifting both fingers cancels transient mode and returns to the bookmarked resolution (e). She can then continue with other standard operations like panning (f).

To activate transient mode, the user presses and holds down a third finger, which saves ("bookmarks") the current location. A shaded circle appears around the pinning finger to make the mode change clear. As long as this pin is held, the system stays in transient mode. Releasing the pin causes the view to snap back to the previous position and resolution. The pinning finger can be moved around the screen so that it doesn't occlude content; the user can continue to use standard gestures and interactions while maintaining the pin. In this implementation, transient can be considered a quasi-mode [13] that can be applied to any subsequent set of operations.

The bookmark is saved at the moment when the third contact point is placed; the user can choose to start in transient mode by ensuring that the third contact is placed before significant movement, or alternately, they can set in the middle of pan/zoom actions to bookmark an intermediate point. Figure 2 shows a more complex example, where standard and transient gestures are intermixed. The use of a quasi-mode in our design makes movement between atomic and transient states fluid.

A simplified state diagram (see Figure 3) illustrates the activation mechanism for this transient quasi-mode, and the transition between transient and atomic operations. A more detailed state diagram to support easy replication of our interaction technique is included in the appendix.

As is common with document and image-editing applications, this implementation makes the assumption that our system utilizes a single-finger to interact, and two fingers to pan and zoom [2].

[1]In some mobile operating systems (e.g. iOS), many applications leverage one-finger pan. We discuss modifications to address one-finger pan in the Discussion section.

[2]We are aware that some applications support one-finger pan and two-finger zoom/rotate; we discuss variants to support this interaction in discussion

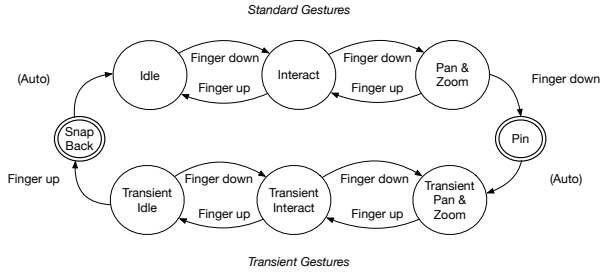

Figure 3. State diagram showing depicting standard (states 0-2) and transient (states 3-5) versions of common operations. Users start in state 0, and transition to different states by adding or removing fingers. Adding a third finger shifts to a transient state, where gestures are interpreted in that context; lifting all fingers returns to the normal interaction state. View interaction is supported in both states.

In determining how to implement transient-mode, we examined the design space for our application to determine which gestures were underutilized, and chose the off-hand pin for activation. In this case, adding a third-finger to the screen, away from the two fingers being used to pan/zoom, causes transient-mode to be activated. The factors used in this case are the number of fingers on-screen, and the distance between the active fingers and the third finger being added. In the absence of a more robust finger-tracking mechanism (using touch-aware screens or computer-vision systems) this mechanism reasonably infers which hand is being used to touch the screen based on proximity. More robust finger-identification mechanisms [15] in commodity hardware might require changes in our design (which we will address further in the Discussion section).

EXPERIMENT

To assess the effectiveness of our revised transient technique, we devised a task which simulated a real-time strategy game interface, where the user has to move between multiple resolutions to solve a task. The goal of this design was to focus on just zoom and pan actions, and minimize the other actions required for task completion.

Task

Our task consists of a 4×3 grid of cells on-screen, with each cell titled A through L. Each of these cells contains a list of number-letter pairs (numbers 1-9, with random letter for each number), all displayed using very small text (6 point font). One of the cells is randomly chosen as the starting cell and highlighted red, and the text is replaced with a target (e.g. "B 7"). See Figure 4 for a high-level view of the task. Note that at the overview level, the text within an individual cell is so small as to be unreadable, necessitating a zoom-in to complete the task.

Figure 4 illustrates the steps. When the task starts, the view is already zoomed-in to the starting cell (a). The goal is to navigate to the cell identified in the starting cell (b), find the value located in the target cell and row (c), return to the starting cell and write it down (d). For instance, in Figure 4, "J 2" in the starting cell indicates that the participant needed to find the value in cell "J", row "2", which is the letter "E"). When

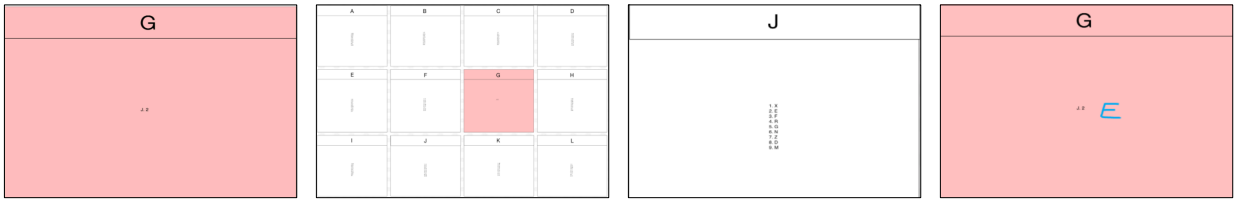

Figure 4. Navigation task. The user is presented with a detailed view of the starting cell, which shows the target cell and row (a). The user needs to zoom-out to locate the target cell (b), zoom-in to read the value (c) and finally return to the starting cell to write down the result (d).

complete, the participants needed to press the "next" button on-screen to move to the next task.

This design forced participants to move from a detail resolution in one cell (Figure 4a), to an overview (Figure 4b), and then down to a detail (Figure 4c) in a different cell to read the data. Next, they had to reverse the set of steps to return to the starting position (Figure 4d). This is similar to the sequence of steps that would be performed when navigating levels in a real-world scenarios. e.g. moving around a PDF document to write notes in the margin, consulting the full author list and title of a reference in a scientific paper, or examining the details of multiple locations on a map in, for example, command-and-control or civilization-style gaming.

Design

We compared our transient zooming technique to both the standard pan-and-zoom, and the omniscient double-tap (or contextual zoom), where the system automatically switches to the correct resolution during the task. These techniques were implemented following these guidelines:

- Normal: implemented as standard two finger pan and pinch-to-zoom, where the midpoint for the zoom action is the midpoint of the two contact points. No acceleration was implemented.
- Double-tap: standard single-point double-tap zooms in and out to the optimal zoom levels (i.e. snapping between overview and detail level for the cell, with cell contents centred). Pan and zoom were disabled for tasks that specified double-tap.
- Transient: a third finger is used to trigger transient mode, but otherwise it performs identically to normal pan and zoom (as described above).

We chose these baselines since standard two-finger pinch-to-zoom and pan is common in multi-touch (especially when a single-stroke is used for interaction such as drawing or writing), so it serves as a reasonable baseline for this comparison. Double-tap can be considered optimal, since it automatically changes to the correct resolution, and centres on the target, removing any guesswork that the user might have to perform. Both approaches are used in commercial systems in support of similar navigation tasks (e.g. tablet applications for reading two-column PDFs, or moving around parts of an image in drawing applications), though double-tap on commercial systems can occasionally select the wrong target resolution.

We considered comparing our technique to accelerated pan-and-zoom techniques [17, 10, 22], but wanted to limit our investigation to explicit, user-controlled gestures.

We designed a within-subject experiment, using technique (normal, double-tap and transient), and starting cell position (A-L, or 12 positions) as conditions, and measuring task duration (ms) and number and type of discrete actions identified by our recognizer (e.g. number of pan vs. zoom actions). Our hypothesis was that duration of the task and number of zoom actions required would be reduced for transient and double-tap compared to standard pan-and-zoom. We expected double-tap to be the fastest technique (since it was designed as an optimal implementation that always guess the correct resolution and eliminates the need for continuous zooming), transient to perform less effectively than double-tap, and standard pan-and-zoom to be the least efficient technique. We did not expect starting cell position to significantly affect performance on any technique.

Each block in our design consisted of 12 tasks (representing the 12 starting positions, randomly presented), repeated across each of the three techniques. When a technique was first introduced, the experimenter demonstrated the optimal use of the technique, answered questions, and then the participant did 5 practice tasks in order to ensure he reaches his best performance for each technique. The order of techniques was counter-balanced across participants to ensure that there was no training effect between techniques (although it is a given that normal and double-tap will already be familiar to most participants). Finally, this was repeated across four blocks to check for learning effects. Each participant completed 12 tasks $\times$ 3 techniques $\times$ 4 blocks = 144 tasks. At the end of the experiment, we asked participants to rate each technique on a 1-5 Likert scale for ease-of-use and effectiveness, and also asked them to rank the techniques by preference.

We recruited 18 participants (mean age 28.9 years (SD=11.0), 7 female, 4 left handed) who where graduate students, working professionals and retirees.

Apparatus

Our software was written in-house in Java 1.8.0, Android 7.1.1 (Nougat) and deployed on a Pixel C tablet. The orientation was locked in landscape, and system controls were restricted so that participants couldn't accidentally exit a task. All data was collected into a log file on the device.

Results

Practice data was recorded, but excluded from our analysis, therefore we collected in total data from $144 \times 18 = 2592$ tasks. Repeated measure ANOVAs are used to determine significance, with Tukey correction applied for post-hoc analysis. We used Shapiro-Wilk to check normality, and for non-normal data that required a non-parametric test, we used Friedman to test significance and Mann-Whitney for post-hoc analysis.

Task Duration

We measured task duration, defined as the time in seconds required to complete each task, starting when the "next" button was pressed to start the task, and finishing when the last stroke of the character was made on-screen. The amount of time spent drawing was minimal (mean 0.44 seconds) and did not significantly vary between techniques, so variations in task duration are attributable to the time spent on navigation.

Tasks were presented in four blocks to each participant. Block has a significant effect on duration ($F_{1,18} = 142.98$, $p < 0.0001$, $\eta_p^2 = 0.54$), with mean task times progressing from 6.12s in block 1 to 4.82s in block 4. This suggests that there is an overall learning effect as participants progressed through the experiment. Task duration decreased across all of the blocks in succession, with significant differences between each block ($p<0.001$). For this reason, we will use block 4 for further analysis, unless otherwise specified.

Technique had a significant effect on task duration ($F_{1,18} = 17.14$, $p < 0.01$, $\eta_p^2 = 0.40$), with transient and double-tap both being faster than standard techniques ($p<0.001$). There is no significant difference between transient and double-tap. Mean task duration was 5.32s (SD 1.72) for normal, 4.73s (SD 1.66) for double-tap and 4.40s (SD 1.58) for transient techniques. For double-tap, this represents a 11.1% improvement over normal. Transient is 17.3% faster than standard pan-and-zoom, but it is not significantly different than double-tap. This is surprising; we designed transient as an improvement over standard pan-and-zoom, but didn't expect it to be comparable to our optimized double-tap (which always selects the exactly correct resolution), given that we are comparing a novel technique to a more familiar and practiced technique.

Actions

We measured the number of pan and zoom actions used to complete each task. Actions are considered to start when the users fingers are placed down, and end when they lift their fingers, or transition to a different gesture by lifting or adding a finger. This means that a user can zoom-out, pan, and zoom-in again as a single continuous gesture, if they don't remove their fingers. This mimics standard pan/zoom behaviour.

We hypothesized that zoom actions would be reduced for double-tap and transient compared to standard pan-and-zoom, but didn't anticipate a reduction in pan actions when comparing techniques (since transient isn't intended to eliminate panning). Pan and zoom counts were not normally distributed, so we used Friedman to test significance, and used Mann-Whitney for post-hoc analysis.

There was no significant difference in pan count when comparing normal and transient techniques. Double-tap didn't allow participants to pan, so it had a constant zero pan count. Transient pan required a mean 3.31 pans, compared to 3.11 pans for standard zoom-and-pan, which represents a 6.4% difference (not significant). This is not unsurprising, since the task encouraged participants to zoom directly on the target.

As expected, there was a significant difference in zoom count by technique (χ^2 (2)=354.77, $p<0.001$). Post-hoc analysis reveals that this is significant for all three techniques ($p<0.001$). Double-tap was a fixed 4.0 (SD 0.0) zoom actions, representing the fixed number of zoom actions that were enforced for that technique. Transient zoom required a 1.43 (SD 0.80) zoom actions, compared to 3.36 (SD 0.92) for standard pinch-to-zoom. Transient shows a 57.6% reduction in the number of zoom actions compared to standard techniques, and a 64.4% reduction compared to double-tap. This is better than expected performance for the transient technique, since participants still need to zoom-out to overview, and zoom-in to the target. However, both standard and transient gestures allow pan/zoom to be performed as a single continuous action, so that participants could (for instance), zoom-out and zoom-in as a single gesture. This method is available for both standard and transient techniques, but may have been leveraged more effectively with transient pan/zoom.

User Feedback

After completing all of the tasks, participants were asked to rate each of the techniques on ease-of-use and usability on a 5-point Likert scale (1-very negative to 5-very positive). The results are summarized in Figure 5.

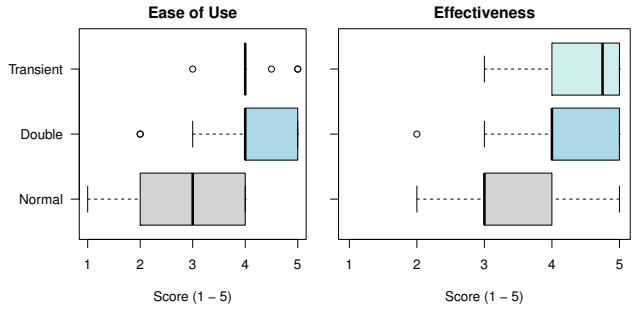

Figure 5. User Ratings of ease-of-use and effectiveness of each techniques on a 5-point Likert scale (1-very negative to 5-very positive)

The ease of use that our participants reported varied significantly by technique (χ^2 (2)=13.46, $p<0.01$). Double tap ($p<0.01$) and transient ($p<0.01$) were both considered to be significantly easier to use than standard pan-and-zoom. There was no significant difference between them, and both were reported at 4.1 on the Likert scale (positive). Standard pan-and-zoom was reported at 2.9 (slightly less than average).

Technique also had a significant effect on the reported effectiveness (χ^2 (2)=16.26, $p<0.001$). Double-tap was not rated as significantly different from either of the other two techniques. Transient was considered to be more effective than standard standard pan-and-zoom ($p<0.01$), but not significantly different

than double-tap. Transient rated at a 4.4, compared to 4.1 for double-tap and 3.3 for standard pan-and-zoom.

Participants were also asked to rank the techniques from most preferred to least preferred. Transient pan-and-zoom were overwhelmingly the preferred techniques; standard pan-and-zoom were the least preferred. See Table 1:

Table 1. Technique Ranking

Rank	1	2	3
Transient	13	4	1
Double-Tap	5	9	4
Normal	0	5	13

When asked to explain their preferences, participants reported that they liked the transient gesture because it was easy to learn, familiar, and similar to techniques that they already knew. "I thought the transient gesture made a lot of sense because it was [built-on standard gestures that] I already knew how to use" (P01) "When I got used to it, I missed the bookmark when I didn't have it. It felt like a normal extension of that technique" (P03).

Also, many participants found the transient technique to be an effective tool for this task. "It was faster to snap back [compared to other techniques]" (P01). "I like this, it's much faster!" (P02). "I liked the transient technique because I'm lazy" (P13).

Many participants also liked double-tap, but found that it felt like "more work"(P14). "Double-tap had too many taps compared to the other techniques. i.e. it takes more actions and costs more energy" (P14). However, for novice users, double-tap may be simpler. One participant, who described herself as a novice, preferred double-tap: "I like this [double-tap], it's the easiest to use, and I already know how to do it" (P09).

DISCUSSION

Our study demonstrates that transient pan-and-zoom are more effective than standard pan-and-zoom, and are at least as effective as double-tap for navigation tasks that require switching between resolutions. Transient techniques reduce the number of zoom actions required in proportion to the amount of zooming required, so more complex tasks will see greater benefit. Double-tap is often effective, but has limitations, since it requires the system designer to anticipate what the user wants to do (i.e. is double-tap zoom-in, or out, and to what level?) Transient pan-and-zoom allows the user to specify their own target resolution and removes this limitation, making it more suitable for ad-hoc interactions.

Furthermore, users prefer transient pan-and-zoom to both standard gestures and double-tap to zoom. They find the technique easy to learn, complementing standard techniques that they already know how to use. This suggests that the transient techniques are approachable for novice to intermediate users, and may serve as an easy transition to expert use [11].

One limitation to our work is the assumption of a specific gesture set. We based our design on common tasks where the user would likely use one finger to interact or draw, and two fingers to perform manipulations like pan and zoom. This is commonly used in tasks like image editing (e.g. Adobe Sketch), or document annotation (e.g. GoodReader) on tablets, where the user intends to manipulate and interact with content. Our implementation assumes no more than two contact points, where we can reliably use a third finger to active the transient quasi-mode.

Moving to a system with a different gesture set with different mappings (e.g. iOS single-finger to pan) may require changes to the activation mechanism, to ensure that there is no collision with existing gestures. For instance, on a system with single-finger pan, does a second finger switch into transient mode, or trigger a zoom? We can use proximity or some other factors to infer if this is a second finger on the same hand, intended to trigger zoom, or a finger on the other hand, intended to trigger transient mode. Alternately, we could use a more robust hand identification mechanism (e.g. computer vision) to determine which hand was used to trigger the second touch. We could also re-leverage a bezel-based activation area for explicit transient activations. BiPad, for example, suggests a number of underutilized gestures that we could employ [23].

FUTURE WORK

Future directions for this work were suggested directly by participants after the study. When asked for ways to improve the transient techniques, some users found the offhand pin awkward, since it sometimes occluded the target (i.e. sliding the pinning finger out of the way required explicit effort). "Sometimes having an extra finger down was awkward" (P06). "I would use my left hand to bookmark, and sometimes it would cover the target, which made it awkward" (P18).

One design consideration might be to implement transient as an explicit mode switch, so that there was no need to hold down a finger to maintain state. However, one of the benefits of using a quasi-mode (i.e. "pinning") is that it makes the state explicit to users, and allows easy layering over existing gestures. Removing the pin might make the technique easier to perform at the cost of increasing the cognitive load, as users need to explicitly keep track of the state.

One other common suggestion was to combine double-tap and transient, to try and gain the benefits of both techniques (i.e. the quick-movement of double-tap in the initial zoom-out and transient to return to the starting state). We suspect that, as with the pilot, the added complexity may make the transient technique less usable, but it would be interesting to find out exactly how far we can extend this approach while still maintaining performance and usability benefits.

Finally, some participants disliked needing to use their offhand to pin. "I don't like using two hands" (P09). "[It's] not as easy coordinating two hands" (P08). Although there are benefits to bimanual interaction with complex tasks [20], unimanual interaction has benefits in specific situations, and could be considered for future designs.

CONCLUSIONS

Standard pan-and-zoom techniques suffer from inefficiencies, and often require excessive repeated transitions when performing tasks that require constant transitions between multiple resolutions and locations in a zoomable interface. We introduce a transient technique that expands the design of standard pan-and-zoom interaction to include the fluid movement between states to reduce the need for repetitive zooming to revisit previous states. In an experiment with 18 participants, we demonstrate that our technique requires 57.6% fewer zoom actions than standard pan-and-zoom, for an overall 17.3% improvement in task completion times. Transient pan-and-zoom is comparable to double-tap in ease-of-use and performance, but is found to be more effective than double-tap and preferred by our participants for navigation tasks.

REFERENCES

1. Pär-Anders Albinsson and Shumin Zhai. 2003. High Precision Touch Screen Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 105–112. DOI:http://dx.doi.org/10.1145/642611.642631
2. Caroline Appert, Olivier Chapuis, Emmanuel Pietriga, and María-Jesús Lobo. 2015. Reciprocal Drag-and-Drop. *ACM Trans. Comput.-Hum. Interact.* 22, 6, Article 29 (Sept. 2015), 36 pages. DOI: http://dx.doi.org/10.1145/2785670
3. Jeff Avery, Mark Choi, Daniel Vogel, and Edward Lank. 2014. Pinch-to-zoom-plus: An Enhanced Pinch-to-zoom That Reduces Clutching and Panning. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 595–604. DOI: http://dx.doi.org/10.1145/2642918.2647352
4. Jeff Avery and Edward Lank. 2016. Surveying Expert-Level Gesture Use and Adoption on Multi-Touch Tablets. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16)*. ACM, New York, NY, USA, 577–581. DOI: http://dx.doi.org/10.1145/2901790.2901895
5. Olivier Chapuis and Nicolas Roussel. 2010. UIMarks: Quick Graphical Interaction with Specific Targets. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology (UIST '10)*. ACM, New York, NY, USA, 173–182. DOI: http://dx.doi.org/10.1145/1866029.1866057
6. Andy Cockburn, Carl Gutwin, Joey Scarr, and Sylvain Malacria. 2014. Supporting Novice to Expert Transitions in User Interfaces. *ACM Comput. Surv.* 47, 2, Article 31 (Nov. 2014), 36 pages. DOI: http://dx.doi.org/10.1145/2659796
7. Alix Goguey, Mathieu Nancel, Géry Casiez, and Daniel Vogel. 2016. The Performance and Preference of Different Fingers and Chords for Pointing, Dragging, and Object Transformation. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4250–4261. DOI: http://dx.doi.org/10.1145/2858036.2858194
8. Yves Guiard. 1987. Asymmetric Division of Labor in Human Skilled Bimanual Action: The Kinematic Chain as a Model. *Journal of Motor Behavior* 19 (1987), 486–517.
9. Eve Hoggan, Miguel Nacenta, Per Ola Kristensson, John Williamson, Antti Oulasvirta, and Anu Lehtiö. 2013. Multi-touch Pinch Gestures: Performance and Ergonomics. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces (ITS '13)*. ACM, New York, NY, USA, 219–222. DOI:http://dx.doi.org/10.1145/2512349.2512817
10. Dominik P. Käser, Maneesh Agrawala, and Mark Pauly. 2011. FingerGlass: Efficient Multiscale Interaction on Multitouch Screens. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 1601–1610. DOI: http://dx.doi.org/10.1145/1978942.1979175
11. Gordon Kurtenbach and William Buxton. 1994. User Learning and Performance with Marking Menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '94)*. ACM, New York, NY, USA, 258–264. DOI: http://dx.doi.org/10.1145/191666.191759
12. Edward Lank and Son Phan. 2004. Focus+Context Sketching on a Pocket PC. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems (CHI EA '04)*. ACM, New York, NY, USA, 1275–1278. DOI: http://dx.doi.org/10.1145/985921.986042
13. Yang Li, Ken Hinckley, Zhiwei Guan, and James A. Landay. 2005. Experimental Analysis of Mode Switching Techniques in Pen-based User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '05)*. ACM, New York, NY, USA, 461–470. DOI: http://dx.doi.org/10.1145/1054972.1055036
14. 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
15. Damien Masson, Alix Goguey, Sylvain Malacria, and Géry Casiez. 2017. WhichFingers: Identifying Fingers on Touch Surfaces and Keyboards Using Vibration Sensors. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 41–48. DOI: http://dx.doi.org/10.1145/3126594.3126619
16. Mathieu Nancel, Daniel Vogel, and Edward Lank. 2015. Clutching Is Not (Necessarily) the Enemy. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 4199–4202. DOI: http://dx.doi.org/10.1145/2702123.2702134

17. Matei Negulescu, Jaime Ruiz, and Edward Lank. 2011. ZoomPointing Revisited: Supporting Mixed-resolution Gesturing on Interactive Surfaces. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS '11)*. ACM, New York, NY, USA, 150–153. DOI: http://dx.doi.org/10.1145/2076354.2076382

18. 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

19. Morgan N. Price, Bill N. Schilit, and Gene Golovchinsky. 1998. XLibris: The Active Reading Machine. In *CHI 98 Conference Summary on Human Factors in Computing Systems (CHI '98)*. ACM, New York, NY, USA, 22–23. DOI:http://dx.doi.org/10.1145/286498.286510

20. Jaime Ruiz, Andrea Bunt, and Edward Lank. 2008. A Model of Non-preferred Hand Mode Switching. In *Proceedings of Graphics Interface 2008 (GI '08)*. Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 49–56. http://dl.acm.org/citation.cfm?id=1375714.1375724

21. Andrew Sears and Ben Shneiderman. 1991. High Precision Touchscreens: Design Strategies and Comparisons with a Mouse. *Int. J. Man-Mach. Stud.* 34, 4 (April 1991), 593–613. DOI: http://dx.doi.org/10.1016/0020-7373(91)90037-8

22. Jessica J. Tran, Shari Trewin, Calvin Swart, Bonnie E. John, and John C. Thomas. 2013. Exploring Pinch and Spread Gestures on Mobile Devices. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '13)*. ACM, New York, NY, USA, 151–160. DOI:http://dx.doi.org/10.1145/2493190.2493221

23. Julie Wagner, Stéphane Huot, and Wendy Mackay. 2012. BiTouch and BiPad: Designing Bimanual Interaction for Hand-held Tablets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2317–2326. DOI: http://dx.doi.org/10.1145/2207676.2208391

APPENDIX

This appendix describes aspects of the system design. We hope this is useful for anyone that wishes to replicate or build on our work.

Implementation

One challenge in designing this system is determining how to implement "standard" features so that they behave in a familiar way. We used the build-in Android double-tap recognizer, to ensure that we used standard and familiar thresholds. Translation and pinch/spread was implemented from scratch, but followed conventions and scaling factors for the platform.

The second major challenge was building a configurable recognizer that could differentiate spurious from intentional actions. At the lowest level, we filtered out small movements (less then 2 pixels, based on pilot data). Movement above this threshold was processed as an action (e.g. ACTION_DOWN, ACTION_MOVE, ACTION_UP in Android). Our gesture recognizer looked for specific pattern of actions, and application context to determine whether a gesture should be triggered. For example, panning gestures were ignored when the application was constrained to work with double-tap input.

Modeling Transient States

We wanted our transient gestures to resemble their standard counterparts (e.g. transient pan compared to standard pan), but recognized that the underlying implementation of these two gestures was quite different. This drove the requirement for explicit transient and standard states. To determine if a gesture was valid, we defined a complete state model for standard and transient modes, with defined transitions between these states (see Figure 6). Adding and removing fingers moves between states, and application state is explicitly saved and restored when we enter or leave the transient state.

We also introduced the notion of wait-states into our design, where the system is waiting for significant movement to transition to a drawing or interactive state. From the user's perspective, this helped retain the "stickiness" of the transient state, and avoid inadvertently switching modes: once in traditional draw or pan/zoom, you need to perform an explicit action to switch to transient mode. Similarly in transient mode, switching between zoom/pan and draw mode requires a user to utilize a default pinned transient state with one contact point.

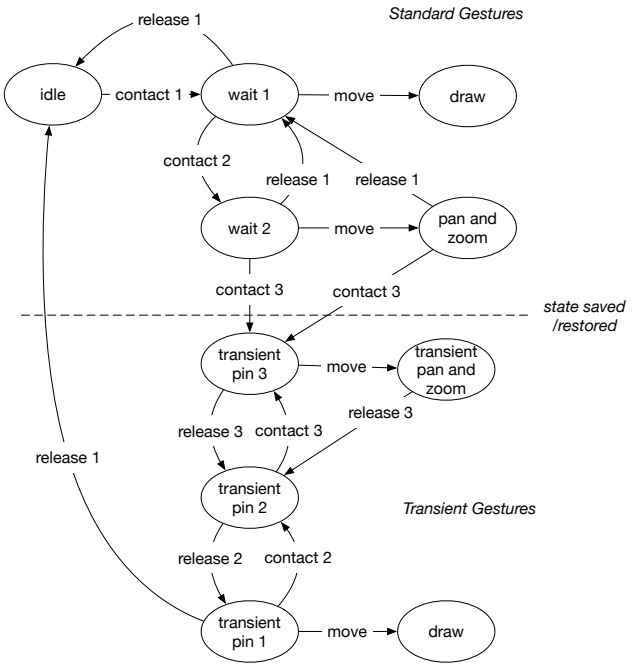

Figure 6. Full state diagram showing idle states and movement triggered transitions. This expands the earlier diagram by explicitly showing non-movement states as pins; pan/zoom are inferred from movement. State is saved when the user moves into a transient state, and restore when moving back to the standard idle state.