How Fast is Fast Enough? A Study of the Effects of Latency in Direct-Touch Pointing Tasks

Ricardo Jota[1], Albert Ng[2], Paul Dietz[2] and Daniel Wigdor[1]

[1]Dept. of Computer Science
University of Toronto,
Toronto, Canada

[2]Microsoft Applied
Sciences Group
Redmond, WA, USA

{jotacosta | dwigdor}@dgp.toronto.edu {v-albeng | pdietz}@microsoft.com

ABSTRACT

Although advances in touchscreen technology have provided us with more precise devices, touchscreens are still laden with latency issues. Common commercial devices present with latency up to 125ms. Although these levels have been shown to impact users' perception of the responsiveness of the system [16], relatively little is known about the impact of latency on the performance of tasks common to direct-touch interfaces, such as direct physical manipulation.

In this paper, we study the effect of latency of a direct-touch pointing device on dragging tasks. Our tests show that user performance decreases as latency increases. We also find that user performance is more severely affected by latency when targets are smaller or farther away. We present a detailed analysis of users' coping mechanisms for latency, and present the results of a follow-up study demonstrating user perception of latency in the land-on phase of the dragging task.

Author Keywords

Latency; direct input; direct manipulations; touch input.

ACM Classification Keywords

H.5.1. Information interfaces and presentation (e.g., HCI): Evaluation/methodology.

INTRODUCTION

Touch-driven interfaces are ubiquitous today, most commonly in the form of mobile phones, but also as tablets, e-readers, and interactive tables. Despite the development of new and better touchscreens, problems with latency—the lag time between a finger touch and the on-screen response—persist, and have been identified as an issue for interactive systems and indirect input devices [13,14,22].

CHI 2013, April 27–May 2, 2013, Paris, France.

It is common for touchscreen devices to exhibit a latency of between 50 and 125ms; a delay that becomes noticeable when users interact with gaming applications, graphics tools, or move objects around a screen [16]. Initial solutions to minimize such effects have been proposed [25], however, the question of *how* and *when* latency affects user performance has not been addressed. Further, the work in this area has focused on *indirect* input devices, where the lagging on-screen cursor is the only representation of current position. This differentiates it from touchscreen interfaces, where the user might choose to ignore cursor position and rely on the finger as a zero-latency indicator of position. This distinction has not been addressed.

Fitts' law is commonly used to model the movement time required to perform basic input tasks for a GUI [7]. Although most formulations of the law do not include a term to explicitly account for latency, Mackenzie and Ware argued that it was inherent in the device specific constants commonly used to tailor the model, and introduced latency as an additional coefficient (see Equation 4) [13]. This formulation was demonstrated to account for latency in the devices the researchers studied. However, this work was limited in that the lowest latency device employed had an input-to-display lag of 8.3ms, and was indirect. Previous research has found that, for direct-touch input, there is a perceptual floor somewhere between 2-11ms, below which users do not notice lag [16]. Thus, 3 open questions remain:

1) *Does latency affect performance with direct-touch systems in the same way that it affects indirect input?*
2) *Is there a floor, below which improvements in response time no longer improve pointing performance?*
3) *What phases of a pointing task are affected by latency, and how do users adjust their behaviors in each phase?*

In this paper, we seek to answer these questions by studying the effect of latency on input to direct-touch devices. Using an experimental prototype, we examine how varying latency between 1 and 50ms can affect the speed and accuracy of input. We present 2 user studies. The first examines the effect of latency on performance in a pointing task, and finds evidence that there is no performance floor to the effect of latency. We also uncover evidence that the effects of latency are tied to particular phases of the pointing task, and thus conduct a second study to measure how users perceive latency for the finger-down portion of the task.

Our research aims to enhance the understanding of how latency affects performance, and, more specifically, how *perception* of latency affects that performance. Ultimately, we aim to contribute to the development of a new generation of touchscreen interfaces that can more accurately mimic the performance of manipulating real-world objects, and to help software architects and interaction designers to focus their efforts on those elements which will yield the best-perceived results.

PREVIOUS WORK

Latency, defined as the "delay between input action and the output response" [13], can be attributed to a number of factors: the data rate of input devices, including mice or tracking devices; time spent processing input and running application logic; and time to render output. The effects of latency on user interaction have been studied by a number of researchers in a number of contexts. We begin by examining the effects of latency on a user's overall perception of an experience. We then examine the effect on performance. Finally, we examine its effects in direct-touch systems in particular.

Latency and Perception

Much of the previous work addresses the issue of *system latency*. Previous studies operate with the assumption of a baseline system latency, and introduce additional latency as a factor. Baseline system latency can be established with techniques such as the one presented by Steed [22].

Latency has been defined as "a cause for reduction of suspension of disbelief." Allision et al. studied the effect of latency in an augmented reality application, noting that latency degrades this illusion of stability and can be seen as a major fault for an interactive system [1]. Nelson et al. reported that latencies of 50 and 100ms were found to affect the ability of participants to visually follow a virtual object with a head-mounted display [15]. Meehan et al. studied the effect of latency at 50 and 90ms on sense of presence in virtual environments [14]. The authors report that an increase in latency significantly contributed to a reduction in sense of presence.

Such research has expanded the understanding of how latency affects the overall experience of a system. However, the focus of our work is on the effects of latency on *performance* of tasks.

Latency and Performance

Researchers have also explored how latency impacts performance when using an indirect input device. So studied the effect of imposing an *additional* latency (0, 55, 110, 220, and 440ms) in a virtual reality environment [20]. Results demonstrated that latency affects pointing performance in this context, and highlighted the correlation between the width of a target and latency. Pavlovych et al. tested mouse input interaction for a targeting task for latencies above 20ms, and found that error rates increase significantly when latency rises above 110ms [17,18,19].

Tasks performed in 3D are also negatively impacted by latency. Teather et al. observed that adding latency to 2 input devices, a mouse and a 3D tracker, significantly affected device performance, and stated that adding 40ms to the system baseline latency affected performance by 15% [23]. Ware et al. studied the effect of latency for reaching tasks in 3D scenarios. Their findings demonstrate that latency (between 70 and 800ms) affects performance [24]. This work found effects which differed from Teather's observations, likely due to the different devices, mice versus 3D trackers—supporting the suggestion that latency effects are device dependent [17]. Ellis et al. found in a pair of works that added latencies between 100 and 500ms significantly degraded performance of path tracing tasks, and that users could distinguish latencies as low as 33ms [5,6].

Our paper builds on this existing work in 2 primary ways. First, we use a touchscreen (direct input) device, where latency may have a different impact on performance than has previously been shown for indirect input devices. Second, earlier work has targeted higher latencies (between 35 and 337ms), whereas we focus latency times in the range of 1 to 50ms, which represents a level of performance beyond today's commercial devices.

Latency in Touch Devices

Unlike indirect input systems which employ a pointing tool such as a mouse, stylus or joystick to provide information to the computer, direct devices (i.e., touch screens, light pens) have no intermediary; the body's movement alone acts as input. As with indirect systems, direct-touch input devices also suffer from the effects of latency. Anderson et al. conducted a qualitative study with users performing various touchscreen tasks (i.e., web browsing, photo viewing and eBook reading), to determine the level of latency users find to be "acceptable" [2]. The authors found that above a delay of 580ms, users found the latency unacceptable, but noted that their experimental tasks were relatively brief, (specifically, zooming, panning, and page-turning), which suggests that latency might be tolerable for longer tasks.

Interestingly, Ng et al. studied the user perception of latency for touch input and produced very different results; for dragging actions with a direct touch device, users were able to detect latency levels as low as 2.38ms [16].

Although these studies provide a critical insight into user perceptions of latency, an understanding of *how* latency affects user performance remains unexplored. Moreover, as touch technology improves, we will soon have the capability to reduce latency to a virtually imperceptible level. Our work explores how touchscreen latency affects performance, and aims to find the latency level at which no further reduction will yield improvements in performance.

Modeling Latency in Pointing

As we have described, MacKenzie and Ware studied the effects of latency between 8.3 and 225ms for target acquisition, and found that lag affects both performance and error rate at all latency levels for mouse-based interactions. MacKenzie and Ware provided a modification to Fitts' law to account for this [13]. In effect, past works have modeled performance, despite the fact that latency is present in most existing pointing tasks due to the computational loop required to provide feedback. Thus, prior work in examining pointing performance has not explicitly accounted for latency, and has generally upheld Fitts' Law as a model for pointing performance. The Shannon formulation for Fitts' law is shown in Equation 1:

$$(1)\quad MT = a + b \times ID$$

Where ID is the "index of difficulty", and is dependent on the distance (D) to a target of a particular width (W):

$$(2)\quad ID = log_2(\frac{D}{W} + 1)$$

MacKenzie and Ware posited, however, that latency is, in fact, an input to Fitts' Law, and that it was being masked in the device-specific constants (a and b in Equation 1). He proposed that latency should be accounted for, and presents a new formulation that takes latency into account:

$$(3)\quad MT = a + (b + c) \times ID$$

Where c is a formulation for latency.

MacKenzie and Ware was able to verify this formulation empirically, and demonstrated how it predicted performance for latencies ranging from 8.3 to 225ms. The present work seeks to extend this prior effort in 2 ways: first, by determining whether these results apply to a *direct touch* input device, where a user could choose to simply ignore latent feedback, and rely on the position of their finger; and second, by determining whether the floor in user perception demonstrated by Ng et al. yields a floor in the effect of latency on pointing performance.

EXPERIMENT 1: POINTING PERFORMANCE

To answer these questions, we asked participants to repeatedly perform pointing tasks on a touch screen display, and included *latency* among the factors in the design of the experiment. This allowed us to observe the effect of latency on performance, and if there is an effect, determine how users changed their behavior to account for latency.

Hypotheses

Our first hypothesis is simply that Fitts' law holds within each level of *latency*. This hypothesis can be viewed primarily as confirming the execution of the pointing experiment, as well as MacKenzie and Ware's model of latency. In our case, varying the device's latency is equivalent to applying Fitts' law experiments in multiple, very similar yet distinct, devices. Thus, for each of those devices, the Fitts' law should still predict performance, albeit with different device-specific coefficients, as per Equation 2.

H1: For each latency level, Fitts' law will predict performance.

Next, we consider whether users will experience the effects of latency, or whether they will simply ignore the latent feedback and rely on cues from hand position. Our overarching expectation is that we will see an effect for latency:

H2: Performance decreases as latency increases.

However, we also believe that this effect will vary depending on the size of the target. With smaller targets, users rely more on feedback to ensure accuracy, thus we expect H3:

H3: Performance is affected according to an interaction between width and latency.

Further, we believe that latency will have an impact according to the distance of the target, as the greater the distance, the greater the cumulative effect of latency (that is, the physical separation between the finger and the cursor). This occurs because the user will "wait and see" to more carefully position the cursor within the target. Thus, we expect H4:

H4: Performance is affected according to an interaction between distance and latency.

Because the ability to lift their finger immediately upon entry to the target is unique to direct-touch input, we anticipate this effect, despite the absence of it in past research.

We also seek to understand if we can identify a floor for the latency effect on performance, below which improvements in latency do not improve performance. Finally, we believe that the effect of latency on performance is actually composed of smaller effects on the various sub-tasks of pointing. In particular, we expect to see an effect for latency in each phase of the task: the time between the user's finger landing on the device and its initial movement; the time spent moving towards the target; and the time spent refining the position of the cursor within the target. We do not have specific hypotheses for these effects, but rather analyzed each of these components separately to understand the user's approach to dealing with latency.

Participants

Forty-Five participants from the local community (28 male, 17 female), ranging in age from 19 to 52, took part in the study. Participants were paid $10 for a one-hour session. All participants were right-handed and had previous experience using touch devices.

Apparatus

Participants executed the experimental task using a High Performance Touch (HPT) prototype similar to the one described in [16] (see Figure 1). The device consists of a 23.5x15.5cm interactive surface, and a programmable FPGA. The prototype was responsible for the trial elements (cursor and target), touch feedback, and generating touch data. The trial logic (from finger touch-down to finger touch-up), was entirely run inside the FPGA, allowing us full control over the experience for latency.

The HPT device was connected to a PC which controlled the flow of the experiment. The computer ran a Java application designed to: issue commands to the prototype to initiate a trial, log the touch data generated by the prototype, and provide user feedback before and after the trial execution. The computer was not responsible for any touch or visual feedback during the execution of the trial.

Task

The dragging task was based on the ISO 9241-9, one-direction tapping task [9], with the modification that participants were dragging an object from 1 on-screen position to another, rather than moving their finger through the air (thus simulating the direct physical manipulation commonly used in touchscreen UIs). This is physically indistinguishable from "pointing" on an indirect trackpad (save clutching). Participants were asked to place their finger upon a cursor, drag the cursor across a path to a target, and release once the cursor was correctly positioned within the target. Targets were displayed as square boxes placed at the same height as the cursor. (see Figure 2).

Procedure

Participants completed a consent form and a questionnaire to collect demographic information. They then received instruction on how to interact with the apparatus, and completed 20 training trials to practice the dragging technique. Finally, they began to complete actual trials.

After execution of each trial, a dialog box appeared to present the result of the trial ('successful' or 'error') and the cumulative error rate (shown as %) for the session (see Figure 2d). Participants were instructed to speed up if the error rate fell below 5%, or slow down if it exceeded 5%. The participant then pressed the spacebar on the PC's keyboard to advance to the next trial. If a target was missed (i.e.: the cursor was not successfully placed into the target), participants repeated the same task until the trial was completed successfully. The procedure lasted approximately 30 minutes and the entire study session was conducted in less than 1 hour.

Design

Dragging tasks varied according to three independent variables: *latency* of the cursor movement (1, 10, 25, and 50ms artificially inserted between input frames); *width* of the target (3, 4, and 5cm; the cursor is a box that measured 2cm, see Figure 2); and the *distance* between the starting position and the target (3.5, 8.5, and 15cm).

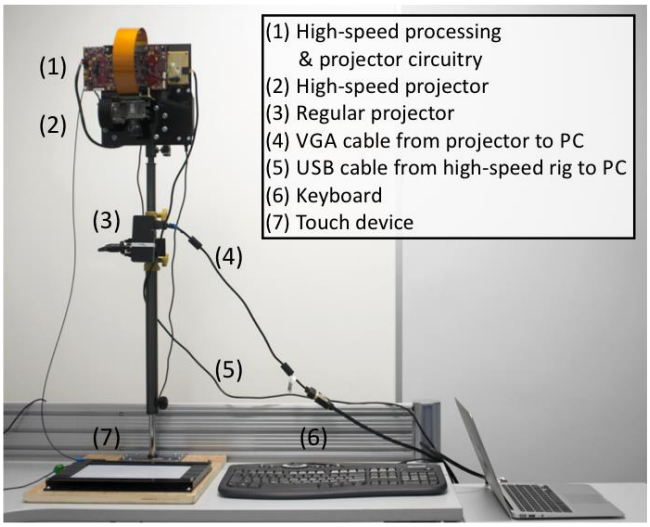

Figure 1. Experimental apparatus.

Each participant performed 8 repetitions of all 36 combinations of levels of *latency*, *width*, and *distance* for a total of 288 trials per participant. The ordering of the 288 trials was randomized across participants. In summary, the design of the experiment can be seen as:

4 levels of latency
3 target widths
3 target distances
8 repetitions
x 45 participants
= 12,960 total trials

Measures and Analysis Methodology

For each trial we captured the total completion time, the location of the cursor for each input frame, as well as the time of arrival and departure of the cursor on touch-down and touch-up. The time between the physical event and the recorded event varied according to the value of *latency*, with a maximum of the time between input frames separating the two (ie, a maximum of 1, 10, 25, or 50 ms), For example, if latency is set at 50ms, then the cursor may appear as much as 50ms after the finger lands on the device. We also measured the number of repetitions (and thus errors) for each trial. The experiment design and accompanying analysis conform to the ISI 9241-9 recommendations [10], as well as the Fitts' study guidelines provided in [21,27,28].

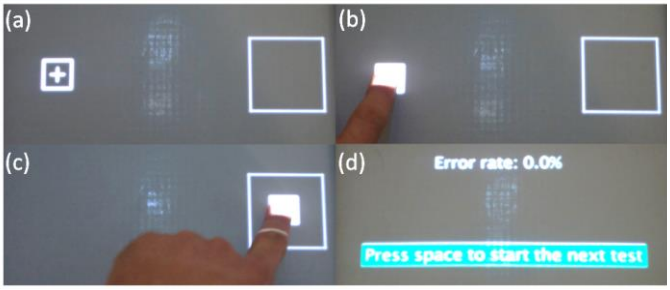

Figure 2. The dragging task for first experiment. Touch device, seen from top-down. (a) before the user selects the cursor. (b) after the cursor is selected. (c) the cursor has been dragged to the target. (d) the screen seen after the user lifts their finger.

RESULTS

We now examine the results of our experiment, grouped by hypothesis. Next, we will examine in more detail the effects of latency on the different phases of the task, in order to allow us to examine the more subtle effects of latency on pointing with a direct-touch input device.

Analysis of Performance Measures

We begin by examining H1. A repeated-measures ANOVA of time values indicates an effect of both width and distance on overall movement time: width ($F_{2, 88}$ = 142.665, p < 0.001); (Distance $F_{2, 88}$ =142.173, p < 0.001). Splitting out the results by latency value, we again find significance for each of distance and width (index of difficulty) (see Table 1). We thus confirm H1, that Fitts' Law continues to assert itself, when considering the HPT at each level of latency as a different device.

To allow us to examine H2, which predicts that performance decreases as latency increases, we compute throughput as the ratio of the index of difficulty and movement time, as taught by [13] (though it should be noted that this practice is not universally accepted [27]). Latency had a significant effect on throughput ($F_{3,132}$ = 114.651, p < 0.001), as shown in Figure 4) and time ($F_{3,132}$ = 45.128, p < 0.001). As is clear in the figure, the effect is manifest as a decrease in throughput as latency increases. We thus confirm H2.

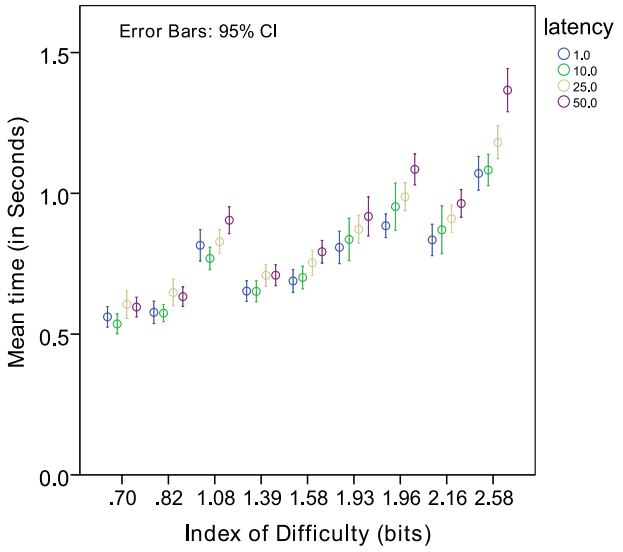

Figure 3. Mean times for ID, separated by latency.

Latency	*Effect of Width on MT*	*Effect of Distance on MT*
1ms	$F_{2, 88}$ = 79.318	$F_{2, 88}$ = 96.384
10ms	$F_{2, 88}$ = 101.406	$F_{2, 88}$ = 70.109
25ms	$F_{2, 88}$ = 71.571	$F_{2, 88}$ = 158.429
50ms	$F_{2, 88}$ = 133.172	$F_{2, 88}$ = 112.773

Table 1. Repeated measures split by latency, all p-values < 0.001.

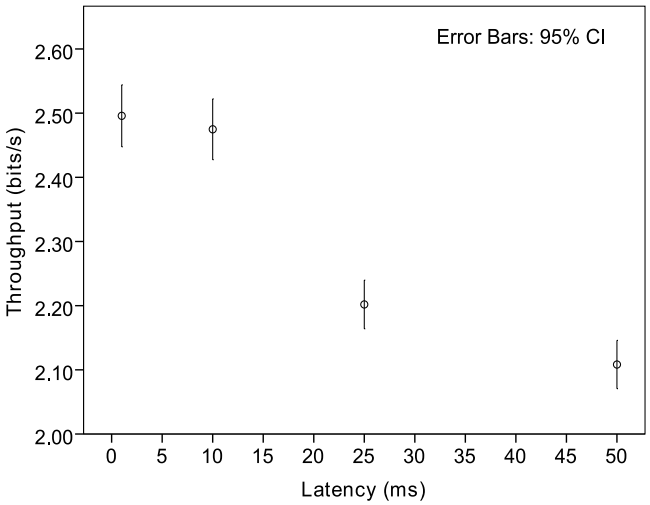

Figure 4. Mean throughput by latency.

Latency had a significant effect on performance, as predicted by H3. In particular, we found a significant effect for the interaction between latency and width on error rate ($F_{6, 264}$=2.581, p=0.019). Although participants made this choice in the speed-accuracy trade-off, we still found a significant effect for target width x latency on movement time ($F_{6,264}$=5.782, p<0.001). We thus confirm H3.

Further, we hypothesized that for sufficiently large targets, a ceiling effect takes place, where the user does not need to "wait and see" if the cursor is positioned correctly within the target, but rather can simply lift their finger as soon as it is centered within the target, even while the cursor is catching up, but visibly inside the target. Indeed, when we limit our analysis to the smallest target size of 3cm, we find that latency is a factor of movement time, ($F_{3,132}$=25.666, p

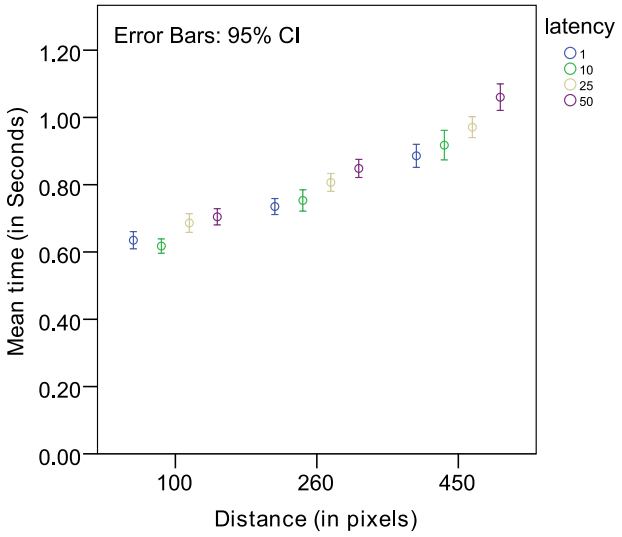

Figure 5. Error rate for each distance, with separate plots for each latency value.

< 0.001), in addition to the previously report effect of errors. This finding led us to believe that latency impacts movement time mostly during the final, refinement stage of pointing, in which the cursor is carefully positioned within the target.

As expected, we found a significant effect for the interaction of distance and latency on movement time ($F_{6,264}$=3.483, p =0.02, see Figure 5). We thus confirm H4.

Floor Effect of Latency on Performance

One of the questions still open is the existence of a floor beneath which pointing performance does not improve. Recall that we were able to confirm H2, and found that latency has a significant effect on throughput. However, *post hoc* pairwise comparisons of particular latency values revealed that there was no significant difference of the effects on throughput between the lowest 2 tested latencies (1ms and 10ms), whereas all other pairs showed significant differences, (see Table 2).

	Throughput		
Latency	Mean Dif.	Std. Error	Sig
1 – 10	*0.007*	*0.010*	*1.000*
1 – 25	**0.063**	0.012	< 0.001
1 – 50	0.109	0.011	< 0.001
10 – 25	0.056	0.009	< 0.001
10 – 50	0.102	0.012	< 0.001
25 - 50	0.046	0.010	< 0.001

Table 2. Pairwise comparisons for latency on throughput, adjusted for multiple comparisons (with Bonferroni correction).

This is not sufficient for us to confirm or deny a floor effect. To perform a deeper analysis of this issue, we consider whether, despite the lack of significance, the mean value for the lowest latency was found to be as predicted, given the trends among the other 3 points.

We performed a linear regression on the results of movement time for the latencies of 10ms, 25ms, and 50ms, omitting the lowest latency value. The results are shown in Figure 6, where it can be seen that this line fits the results precisely (R^2=0.956). Using this line to predict the mean movement time for latency=1ms, we find an expected mean of 747.6ms. Our empirically-measured results report a mean of 751.3ms, well within the 95% confidence interval predicted. This evidence suggests there may not be a performance floor.

This observation indicates that any increase in latency decrease performance. To further understand this effect, we conducted a more detailed analysis of the movement logs of our study.

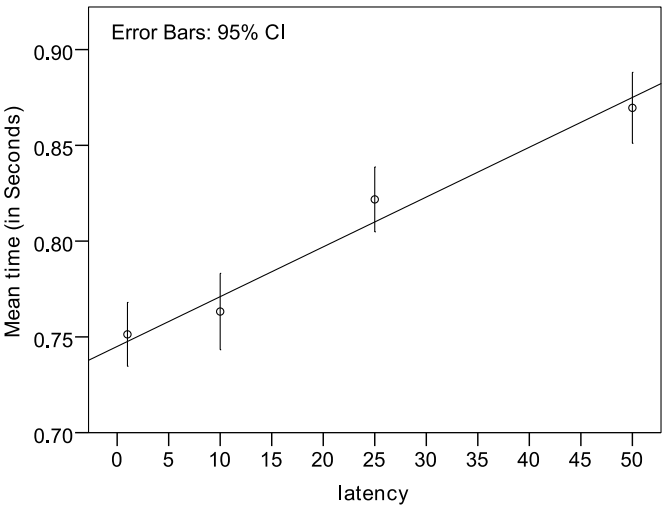

Figure 6. Regression line for predicted mean times.

Users' Response to Latency

Although an aggregate of user behavior is indicated in the performance measures, we wished to further examine precisely *when* in the pointing task the effects of latency affect performance. To that end, we analyzed our logs of cursor position throughout each pointing task.

We divided data for each trial event into three phases, described as follows:

$\underline{Phase_i}$ is the time from touch-down to the beginning of finger movement. It can be seen as the user's initial reaction time to the cursor indicating to them that their finger has been sensed, and they may begin to drag. To account for jitter, a 5px threshold was set on the initial movement.

$\underline{Phase_m}$ is the time from the beginning of finger movement to the entry of the finger into the target zone. This roughly corresponds to the ballistic phase of movement described in traditional pointing models [12], but is distinct in this case: given the difference in feedback between their finger and the cursor, the user may, or may not, treat this phase as an open loop movement. We have not attempted to detect whether or not the movement is open loop, but rather simply whether they are close enough (or not) to the target that there is value in conducting a closed-loop refinement. We are interested to know whether, during this phase, users may opt to slow down, perhaps to maximize the correspondence between finger and cursor positions.

$\underline{Phase_f}$ is the time spent with the finger inside the target area. This time roughly corresponds to the feedback-adjusted final adjustments described in previous works [12]. However, again due to direct input, the user may be able to gain some efficiencies by fine-tuning their finger position prior to the 'arrival' of the latent cursor.

Figure 7 shows the mean values for time spent at each phase of the experiment, broken out for each latency level. We consider first the movement phase, $Phase_m$, where we see a significant effect for latency on time spent ($F_{3,129}$ =

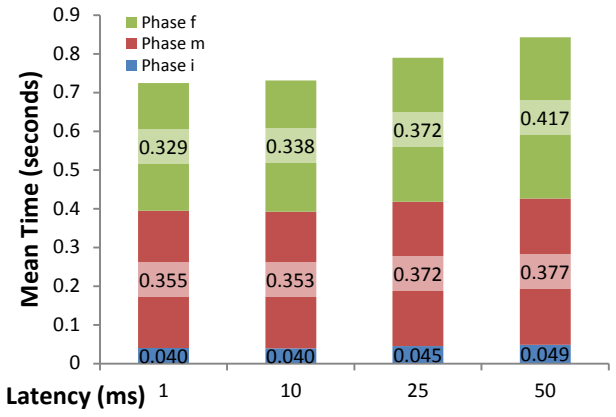

Figure 7. Mean lengths of each of the 3 phases of the task, per latency level.

8.845, p<0.001). *Post hoc* pairwise comparisons of $Phase_m$ show that the difference between the 1ms and 10ms and the 25 and 50ms are not significant. There is significance between 10 and both 25ms and 50ms. Compared to 10 ms, we do see greater time spent in $Phase_m$ for 25ms and 50ms conditions, suggesting that users may slow down somewhat when the cursor trails behind their finger (mean 6% increase between 10ms and 50ms conditions).

In $Phase_f$, when the finger is within the target, we see the greatest difference in time across latency values ($F_{3,129}$ = 63.741, p<0.001). *Post hoc* pairwise comparisons find levels of the length of $Phase_f$ at 1ms and 10ms are not significantly different from one another (p=1.0), however, 25ms and 50ms are each significantly different from 10ms. This difference is likely due to latency's effects as users perform fine adjustments, positioning the cursor within the bounds of the target. Between 10ms and 50ms latencies, we see a 26% increase in mean time spent in the movement phase.

In further analyzing the logs, we found individual differences in behavior within each of $Phase_m$ and $Phase_f$ under varying latency. In general, 3 distinct strategies were observed. Some participants would quickly move their finger to the target, and, under high-latency conditions, wait for the cursor to catch-up before performing corrective movements. This would minimize $Phase_m$, but prolong $Phase_f$. Alternatively, some participants tended to slow down during movement, apparently to maximize the correspondence of the cursor and finger. This strategy increases the time spent in $Phase_m$, while eliminating the "waiting" component of $Phase_f$. Finally, some participants would apparently "split the difference", slowing down somewhat in $Phase_m$, but not maximize correspondence.

Finally, we consider the first phase, $Phase_i$, which can be characterized as the user's reaction time to the system's feedback that their touch had been registered. Here we find that latency had a significant effect on time spent in the phase ($F_{3,129}$=21.795, p < 0.001). It is interesting to note, however, the lack of difference between 1ms and 10ms of latency. If participants were able to benefit from the improved performance, we would expect the mean of this difference to be ½ of a frame of input (4.5ms).

This result suggests that participants are unable to perceive the difference between 1ms and 10ms of lag between the time their finger lands on the screen and when the result is displayed. This also suggests, contrary to prior thinking, that they *are* able to perceive the difference between 50ms and 1ms. Because this result is unexpected, it led us to conduct a second experiment, designed to detect the threshold value at which users are able to detect latency for the land-on portion of a dragging task.

EXPERIMENT 2: PERCEPTION RESPONSIVENESS

Earlier work by Ng et al. demonstrated that users are able to perceive as little as 2.38ms of latency when dragging an object [16]. As they point out, however, it is likely that participants were actually noting the disparity between the position of the finger and the cursor, caused by latency, rather than the latency itself. Further, their experiment, did not examine users' ability to perceive differences in latency for the *land-on* portion of the task. Earlier work suggested that perception of latency for this type of event is limited to 100ms and above [4]. However, our results in Experiment 1 suggest that users may be able to perceive latencies below 50ms. To investigate this further, we conducted an experiment in which participants were shown feedback in response to touching, in which we modulated the latency between finger-down and the appearance of the feedback. Our goal was to determine the precise lower bound at which latency can be detected by a user when first touching their finger to the screen.

Like Ng et al., we measured perception using the just-noticeable difference (JND) experimental method [16]. The JND is defined as the threshold level at which a participant is able to discriminate between two unequal stimuli: one consistently presented at the same level, termed the *reference*; and one whose value changes dynamically throughout the experiment, termed the *probe*. A commonly accepted value for the JND is the lowest probe value at which a participant can correctly identify the reference 75% of the time. Any *probe* value that cannot be distinguished from the *reference* value with this level of accuracy is deemed "not noticeably different" from the reference [11].

We conducted a within-subjects experiment to determine the JND level of the probe, when compared to our reference (1ms). While such a determination does not provide an absolute value for the lowest perceptible latency, it does serve as our "best case" floor condition against which we are able to measure other levels of latency. This experiment was conducted as a second part of our first experiment; participants would first finish Experiment 1, and then complete Experiment 2. This ensured that participants had ample time to familiarize themselves with the concept of latency, and to become facile with the apparatus.

Apparatus
The hardware used was the same as in the first experiment. In software, the low-speed projector was used to render a target on-screen (Figure 8, left). When the user touched the target, the high-speed device displayed a rectangle around their finger (Figure 8, right). The time between the touch and the display of the high-speed feedback was modulated experimentally.

Participants
Twenty participants of Experiment 1 were asked to continue and complete Experiment 2. Participants were paid an additional $10 for this 30-minute session.

Task and Procedure
Participants were asked to tap the on-screen target, and to note the latency of the appearance of the rectangle. The latency value was then reset, and participants were asked to touch the target and note latency again. They were then asked to indicate which of the 2 trials was "faster". Participants could tap repeatedly, and were also allowed to repeat either of the 2 latency values. Participants indicated their choice verbally to the experimenter, and the next latency pair was loaded.

Design
In order for each trial to converge at our desired JND confidence level of 75%, the amount of added latency was controlled according to an adaptive staircase algorithm, [11, 16]. Each correct identification of the reference value caused a decrease in the amount of latency in the probe, while each incorrect response caused the probe's latency to increase. Step sizes followed the *simple weighted up-down method* described by Kaernbach, wherein increases had a three-fold multiplier applied to the base step size, and decreases were the base step size (initially 8ms) [10]. When a participant responded incorrectly after a correct response, or correctly after an incorrect response, this was termed a "reversal" as it caused the direction of the staircase (increasing or decreasing) to reverse. The step size, initially 8ms, was halved at each reversal, to a minimum step size of 1ms. This continued until a total of 10 reversals occurred, resulting in a convergence at 75% correctness (per [10]).

Each participant completed two staircase "runs". One run started at the minimum probe latency (1ms) and the other at the maximum (120ms). The higher start value (120ms) was chosen because pilot testing made it clear that this value would be differentiated from the 1ms reference with near 100% accuracy, avoiding ceiling effects.

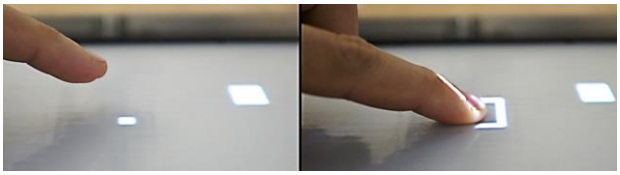

Figure 8. Software for Experiment 2. Left: before the participant touches. Right: after he has touched.

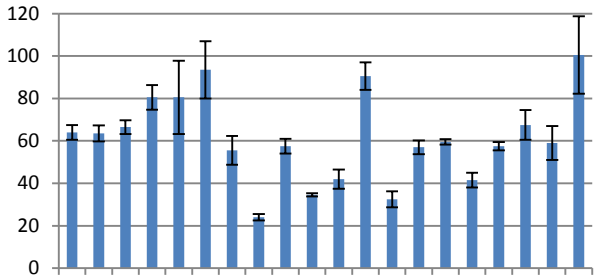

Figure 9. Mean JND (ms) per participant. Bars are std-dev.

Staircases were run in parallel, in interleaved pairs to prevent response biases that would otherwise be caused by the participants' ability to track their progress between successive stimuli [10]. The entire experiment, including breaks between staircases, was completed by each participant within a single half-hour session.

Results and Discussion
Our results show that participants' ability to perceive latency in feedback to the land-on event range from 20 to 100ms, with a mean JND of 64ms (standard deviation of 24ms, see Figure 9). This helps to explain the floor effect observed for $Phase_f$ in Experiment 1—because participants can only perceive touch feedback above 20ms, they are unable to react more quickly to the feedback on finger-down under 1ms of latency than they are under 10ms of latency.

DISCUSSION AND FUTURE WORK
We divide our discussion into 2 sections. The first discussion is intended to establish guidelines as to what level of performance is required to minimize various issues caused by latency, while the second introduces considerations on how to adapt the user interface to take advantage of a low latency system.

How Fast is Fast Enough?
As was described by Ng et al., achieving latencies even within an order of magnitude of those we have investigated here, may well require fundamental changes to software architectures [16]. It is perhaps heartening that, although users may be able to perceive latencies below 10ms, there is apparently little performance benefit of achieving such speeds. Indeed, the performance on direct physical manipulation tasks appears to be little advantaged by latencies much lower than 25ms. Further, when we limit physical interaction primitives to tap-based input alone, and compare 50ms and 1ms of latency, it would appear that a benefit only on the order of 7% improvement in response times to touches is produced. Indeed, if input were restricted to tapping, most users appear to not even notice improvements in latency below 40ms.

Thus, there is a tension between the common desire to employ direct physical manipulation metaphors, and the desire to hide latency from the user. Past efforts in this regard have focused on altering visual responses, so as to

"trick" the user into perceiving the separation between the finger and the feedback of finger location as intentional design elements [26]. This may help to explain why developers of modern touch devices, such as the iOS and Android, have so frequently opted for tap-based interaction primitives, despite their many disadvantages when compared with primitives that utilize some dragging to actuate [26].

In summation, it would appear that achieving a touch-to-display latency of 20ms would be sufficient to ensure the majority of users do not perceive it, and thus would see no performance degradation, when input is restricted to tapping. When direct manipulation is employed, latencies down to 2.38ms are required to alleviate user perception when dragging (per [16]). Our results are consistent with this, in that our results suggest that performance continues to be improved when reducing latencies, and suggest that improvements beyond this level may continue to aid performance of common tasks. Such improvements could contribute to the adoption of tablets beyond the niche of media consumption. One example is the accurate representation of signatures that require precise, feedback with minimal latency.

User Interface Implications

Our prototype did not have any other source of latency other than the input latency artificially inserted by the experimental setup. Often, this is not the case: most real world operating systems, windowing and UI toolkits, and applications, introduce additional, often widely variable, latency. Ng et al. proposed overcoming this by bifurcating the input stream, so that all touch events are simultaneously routed to a high-speed, hardware-based, low-latency subsystem, and a copy is sent to the input stack for regular processing [16]. Unaddressed, however, is how to modify the software stack, given our knowledge of users' perceptual abilities with respect to latency.

One commonly employed software tool is an input event queue. Current implementations of the input queue already account for latency and skip events that are too old, or combine events that are identical (or otherwise provide a null aggregate affect). What our results show is that responding to position events mid-drag which are older than 10ms will reduce performance.

A naïve approach would be to simply discard events beyond a threshold age. In this case, the temptation might be to flatten queue events within the last 2ms—or even just to ensure only 1 dragging event is ever queued—to reduce the perception of latency. Such an approach would, however, be prone to jitter whenever the system could not maintain the high refresh rate. This trade-off is not desirable [19]. A common approach to reduce jitter is to animate transitions between locations, but animation would, by its very nature, reintroduce lag into the system.

An alternative we are exploring is to, indeed, flatten the queue, so that intermediate events are always discarded. However, this change requires the queue management algorithm to know additional details about the state of the various UI widgets (indeed, that something is being 'dragged', and thus that the path does not matter). To reduce the perception of jitter, we are exploring the adaptation of Phosphor, which provides annotations in place of animations. This has the advantage of showing the user the path of movement, without the slow-down necessitated by animation [3].

Another implication of our findings is target size. Past work on target size has focused on land-on or take-off target size, but has not explored the importance of dragging target size. Despite this lack of attention, dragging to small targets is indeed common on touch devices; for example, moving the text carrot to position it at the beginning of a word. Our results demonstrate that small targets are harder to hit under high latency. To solve this, designers can focus on increasing the size of targets for drag operations, or indeed explore temporarily, dynamically changing target size according to input latency, thus slightly decreasing index of difficulty to facilitate targeting. In addition, practitioners can, if possible, increase input event dequeuing frequency whenever the user closes-in on a small target, perhaps by prioritizing the feedback thread.

Today, designers compensate for latency through the use of animated effects, such as the springiness employed in iOS. As latencies are reduced, such effects become superfluous. Thus, an interesting area of future work is to examine how current primitives are affected by latency, and how their design, look, and feel may be changed.

Finally, our results suggest that there is a specific window of opportunity between the time he user's finger lands on the screen, and when feedback for that event must take-place to be perceived as instantaneous. Although not as long as previously thought, it is still sufficiently long to be utilized to perform operations such as waiting for the finger to make full contact, and mapping input to a point [8]. The size of this window was found in our experiment to be somewhere between 20ms (100 % of participants) and 40ms (85% of participants).

CONCLUSION

In this paper, we study of the effect of latency in direct-touch pointing tasks. We conducted two user studies: the first measured performance under various latency conditions, while the second measured user perception for initial input feedback. We found that latency was a factor that influences both performance and perception. As past results predicted, users' performance decreased with increasing latency. We also found that it is the final, refinement stage of the pointing task which is most affected by input latency.

We also revealed that users' reaction times are limited, and thus, that providing feedback of a user's initial touch more quickly than 10ms does not appear to improve performance. Indeed, we found that no participant could discern latency for land-on below 20ms, and 85% could not differentiate between 40ms and 1ms of latency for the land-on event.

Thus, we advocate a focus on tapping input for higher-latency hardware, and to ensure any hardware intended to rely extensively on direct physical manipulation metaphors aim for latency approaching 1ms. We believe that this will improve user performance and create a suspension of disbelief on direct manipulation metaphors applied today.

ACKNOWLEDGEMENTS

We would like to thank Steven Sanders and Kim Wright for their input on the document, and all participants for their zeal during the experiments.

REFERENCES

1. Allison, R. S., Harris, L. R., Jenkin, M., Jasiobedzka, U., and Zacher, J. E. (2001). Tolerance of Temporal Delay in Virtual Environments. IEEE VR 2001, 247–254.
2. Anderson, G., Doherty, R., and Ganapathy, S. (2011). User Per-ception of Touch Screen Latency. Design, User Experience, and Usability. Theory, Methods, Tools, and Practice. 195-202.
3. Baudisch P., Tan D., Collomb M., Robbins D., Hinckley K., Agrawala M., Zhao S., and Ramos G. (2006). Phosphor: explaining transitions in the user interface using afterglow effects. User interface software and technology (UIST '06). ACM, New York, USA, 169-178.
4. Card, S. K., Mackinlay, J. D., and Robertson, G. G. (1991). The Information Visualizer: An Information Workspace. Proceedings of (CHI '91), 181-188.
5. Ellis, S. R., Breant, F., Manges, B., Jacoby, R., and Adelstein, B. D. (1997). Factors influencing operator interaction with virtual objects viewed via head-mounted see-through displays: viewing conditions and rendering latency. Virtual Reality 1997, 138–145.
6. Ellis, S. R., Young, M. J., Adelstein, B. D., and Ehrlich, S. M. (1999). Discrimination of changes of latency during voluntary hand movements of virtual objects. Human Factors and Ergonomics Society, 1182–1186.
7. Fitts P. (1954). The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology, volume 47, number 6, June 1954.
8. Holz C. and Baudisch P. (2011). Understanding touch. In Proceedings of the 2011 annual conference on Human factors in computing systems (CHI '11). ACM, New York, NY, USA, 2501-2510., 91–139.
9. ISO/DIS9241–9 Ergonomic requirements for office work with visual display terminals (VDTs) – Part 9: Requirements for non-keyboard input devices. International Standard, International Organization for Standardization, 2000.
10. Kaernbach, C. (1991). Simple adaptive testing with the weighted up-down method, Perception & Psychophysics 49, 227-229.
11. Levitt, H. (1971). Transformed up-down methods in psychoa-coustics. Journal of the Acoustical Society of America, 49(2): 467–477.
12. MacKenzie, I. S. (1992). Fitts' law as a research and design tool in human-computer interaction.
13. MacKenzie, I. S., and Ware, C. (1993). Lag as a determinant of human performance in interactive systems. ACM CHI 1993.
14. Meehan, M., Razzaque, S., Whitton, M. C., and Brooks, F. P. (2003). Effect of Latency on Presence in Stressful Virtual Environments. In IEEE VR 2003, 141–138.
15. Nelson, W. T. and Roe, Merry M. and Bolia, Robert S. and Morley, Rebecca M. (1998). Assessing Simulator Sickness in a See-Through HMD: Effects of Time Delay, Time on Task, and Task Complexity. Proceedings of the 2000 IMAGE Conference.
16. Ng, A., Lepinski, J., Wigdor, D., Sanders, S., Dietz, P., Designing for Low-Latency Direct-Touch Input. To appear in Proceedings of the ACM symposium on user interface software and technology. ACM, New York, NY, USA. (in press).
17. Pavlovych A., Gutwin C. (2012). Assessing Target Acquisition and Tracking Performance for Moving Targets in the Presence of Latency and Jitter, Graphics Interface 2012, May 2012.
18. Pavlovych A., Stuerzlinger W. (2011), Target Following Performance in the Presence of Latency, Jitter, and Signal Dropouts, Graphics Interface 2011, 33-40, May 2011.
19. Pavlovych A., Stuerzlinger W. (2009). The Tradeoff between Spatial Jitter and Latency in Pointing Tasks, ACM Symposium on Engineering Interactive Computing Systems 2009, 187-196, July 2009.
20. So, R. H. Y., and Chung, G. K. M. (2005). Sensory Motor Responses in Virtual Environments: Studying the Effects of Image Latencies for Target-directed Hand Movement. In Engineering in Medicine and Biology Society, IEEE-EMBS 2005, 5006–5008.
21. Soukoreff W. and MacKenzie I. (2004). Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI. International Journal of Human-Computer Studies. 61, 6 (December 2004), 751-789.
22. Steed A. (2008). A simple method for estimating the latency of interactive, real-time graphics simulations. In Proceedings of the 2008 ACM symposium on Virtual reality software and technology (VRST '08). ACM, New York, NY, USA, 123-129.
23. Teather, R., Pavlovych, A., Stuerzlinger, W., and MacKenzie, S. (2009). Effects of tracking technology, latency, and spatial jitter on object movement, IEEE 3DUI 2009, 43–50.
24. Ware C. and Balakrishnan R. (1994). Reaching for objects in VR displays: lag and frame rate. ACM Trans. Comput.-Hum. Interact. 1, 4, 331-356.
25. Wigdor D., Williams S., Cronin M., Levy R., White K., Mazeev M., and Benko H. (2009). Ripples: utilizing per-contact visualizations to improve user interaction with touch displays. In Proceedings of the 22nd annual ACM symposium on User interface software and technology (UIST '09). ACM, New York, NY, USA, 3-12.
26. Wigdor, D., Wixon, D. (2011). *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture* (1st ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
27. Zhai, S. Characterizing computer input with Fitts' law parameters ―The information and non-information aspects of pointing. International Journal of Human-Computer Studies, 61(6), 791-809. December 2004.
28. Zhai, S., Kong, J., Ren, X., Speed-accuracy trade-off in Fitts' law tasks — On the equivalency of actual and nominal pointing precision. International Journal of Human-Computer Studies, 61(6), December 2004.