# Measuring the Drag Latency of Touchscreen Displays for Human-in-the-Loop Simulator Experiments

Vrouwenvelder, S.C.F.; Postema, F.N.; Pool, D.M.

**DOI**
10.2514/6.2021-0896

**Publication date**
2021

**Document Version**
Final published version

**Published in**
AIAA Scitech 2021 Forum

AIAA SciTech Forum
11–15 & 19–21 January 2021, VIRTUAL EVENT
AIAA Scitech 2021 Forum

10.2514/6.2021-0896



# Measuring the Drag Latency of Touchscreen Displays for Human-in-the-Loop Simulator Experiments

Simon Vrouwenvelder*, Ferdinand M. Postema†, and Daan M. Pool‡
*Delft University of Technology, Delft, The Netherlands*

**This paper presents a novel methodology for measuring the drag latency of touchscreens. Drag latency is a characteristic of touch hardware that can affect key metrics of human performance as typically collected in human-in-the-loop (simulator) experiments. The proposed methodology uses a laser and laser sensor to obtain an external measurement of stylus position that is compared, in time, with the digitally measured touch event data. The drag latency is estimated as the time shift required to align the locations of touch sensor activation with the recorded laser beam crossings. Application of this methodology on two touchscreens from different vendors showed that touchscreen drag latency is in general strongly dependent on the input speed and varies between 28 (fast inputs) and 200 ms (slow inputs) for the tested hardware. Furthermore, while drag-latency characteristics seem to generally follow an exponential decay curve as a function of input speed, the latency was found to differ by a factor 2 at low input speeds between both touchscreens. Using the measurements from our proposed methodology to obtain predictive models of touchscreens' drag-latency dependency on input speed was found to facilitate accurate compensation for the drag latency for three different input tasks (chirp, multi-sine, and step) relevant to human-in-the-loop testing. The results suggest that measuring, and compensating for, touchscreen drag latency is essential for human-in-the-loop experiments, to ensure consistent measures of human performance and to enable replication of measured effects.**



## I. Introduction

With increasing use of touchscreen technology in aviation and future cockpit developments, currently also an increasing number of experimental human-in-the-loop studies focus on touch input devices [1–6]. For repeatable experiments and verifiable results, it is essential that the detailed characteristics of human-in-the-loop experiment hardware are known (and reported). In our experience, this is a currently underdeveloped aspect of most touchscreen-related research, also as detailed knowledge of key characteristics such as input latency is generally not available for most commercial off-the-shelve (COTS) touchscreens.

The touchscreens that are typically used for human-in-the-loop experimentation will generally differ in two main ways. The first is the screen's coating, which may increase or decrease friction with the stylus or finger and provides an either matte of glossy top layer, affecting glare. The main, and more hidden, difference between touchscreens, however, are device-specific input delays as for instance incurred in low-level touch sensor data processing. For human-in-the-loop experiments, detailed knowledge of such hardware latencies is essential, as otherwise they are (falsely) attributed to the input behavior of experiment participants. For touchscreens a distinction is generally made between different 'categories' of delays [7]: 1) *tap latency* (separately for touch and release events), 2) *drag latency*, 3) *initial movement latency*, and 4) *display latency*.

*Tap latency* is defined as the time difference between a physical tap on the screen and a registered response (touch or release). *Initial movement latency* is the time it takes for a user to confirm that his or her response has been registered when dragging an object across the screen, while *display latency* is not related to the touch capacity but the delay in the screen image presentation that can, for example, be measured by the method of [8]. The tap latency is generally not the same as the latency experienced when performing a dragging operation on a touchscreen, due to the fact that touch measurements are subjected to a measurement noise effect often referred to as *jitter* [9, 10]. In most touch sensors, jitter

*M.Sc. student, Control & Simulation Section, Faculty of Aerospace Engineering, P.O. Box 5058, 2600GB Delft, The Netherlands.
†Instrumentation Engineer, Control & Simulation Section, Faculty of Aerospace Engineering, P.O. Box 5058, 2600GB Delft, The Netherlands; f.n.postema@tudelft.nl.
‡Assistant Professor, Control & Simulation Section, Faculty of Aerospace Engineering, P.O. Box 5058, 2600GB Delft, The Netherlands; d.m.pool@tudelft.nl. Senior Member AIAA.

is reduced by averaging inputs over multiple measurement frames, which introduces an additional source of latency. This is referred to as the *drag latency*. As explained in [7, 11], in the integrated circuits that process the raw touch sensor data, slow touch movements are commonly chosen to require better accuracy (i.e., improved noise suppression) than faster movements, which introduces a dependency on the characteristics of the input signal.

Especially touchscreen drag latency has been found to be a critical factor in continuous input tasks, such as tracking tasks [4]. Average reported values lie between 50 and 200 ms [12, 13], with the minimum noticeable drag latency being 6.04 ms ($\sigma$ = 4.33 ms, ranging between 2.38 and 11.36 ms) [12, 14]. How exactly human operators respond to drag latency is a different question. According to [14]: *"[touchscreen/drag] latency [is] a factor that influences both performance and perception"*, see also [9, 15]. In other words, drag latency increases movement times while decreasing the information throughput, i.e., reduced task performance. In addition, a moderate effect on input rates has been reported [9].

With touchscreen drag latencies that depend on the speed of finger movements across the screen, as well as design choices made by touch hardware manufacturers in low-level touch sensor data processing, it is essential for valid human-in-the-loop experimentation that more insight into the latency characteristics of touchscreens is obtained. For example, in a recent experiment where the same dragging task was performed on two different touchscreens [4] a notable difference in task performance was observed. For this reason, this paper proposes a novel methodology for measuring the drag latency of touchscreens for a wide range of input speeds. Our proposed approach makes use of an external screen position measurement obtained from a laser beam that is occluded by the stylus used for touch inputs. This paper will present both the full details of the measurement setup and required data processing to obtain an estimate of the drag latency. In addition, the methodology is applied to two different touchscreens that have previously been used at our institute for human-in-the-loop experiments [4, 5]: a Dell P2314T (hereafter referred to as 'Screen 1') and an Iiyama ProLite TF1534MC ('Screen 2').

This paper is structured as follows. First, Section II describes our proposed method for measuring touchscreen drag latency, including the estimation of the drag latency from recorded measurement data. Section III presents the results of how drag latency is found to vary with input speed for the two tested screens, as well as additional measurements taken to verify and validate our proposed approach. The conclusions are presented in Section IV.

## II. Method

### A. Latency Measurement

This paper describes a custom setup developed to measure the latency in touchscreens' processing of dragging inputs. This latency was measured by comparing the physical position of a stylus (externally measured using a laser relay) to the digital touch event locations processed by the operating system's (Linux) kernel. This approach is graphically illustrated in the left figure in Fig. 1, which shows a laser and a laser sensor (black boxes) mounted on opposite sides of a touchscreen. When a stylus is dragged across the touchscreen this results in light-to-dark and dark-to-light events being registered by the laser sensor. In this paper we use circular dragging input paths across the screen surface, which ensures an as-constant-as possible input velocity, given human limitations that favor a certain direction [16, 17]. A range of different angular velocities was tested to explicitly measure the input-velocity dependency of the drag latency.

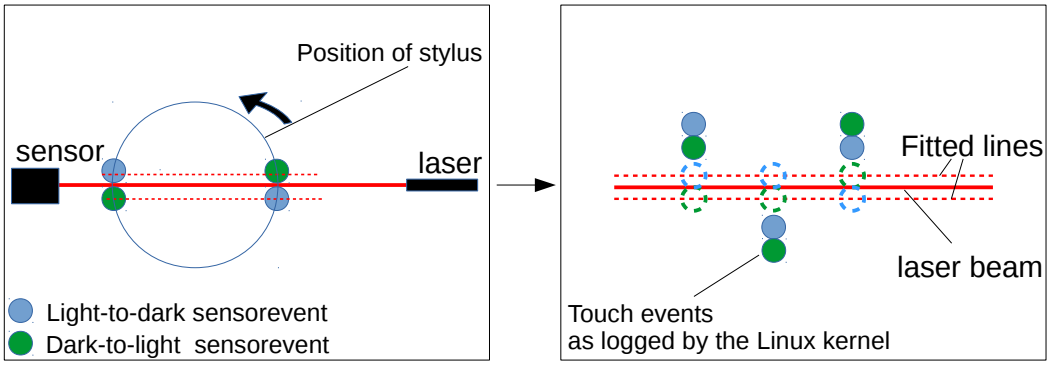



**Fig. 1 Schematic working principle of the latency measurement experiment. Left is the physical set-up, right is a graphical illustration of the software set-up used to compute the shift in time required (latency) for digital measurements to physically line up.**

With the approach shown in Fig. 1 at left, rather than measuring the absolute screen position, the physical position of the stylus relative to the laser beam was inferred from the occlusion of the laser sensor. As presented in the right figure of Fig. 1, the kernel touch events *initiated* above and below the laser beam *should* physically be on a single line. However, due to drag latency and the input pattern, the respective touch events at the times of light-to-dark or dark-to-light laser events are generally not aligned. The drag latency can therefore be estimated by shifting these touch events in time to fit two offset lines parallel to the laser beam. Note that due to the fact we consider a circular input pattern the laser beam is crossed both from both directions. Thus, both the 'top' and 'bottom' fitted lines are determined from both light-to-dark (blue) and dark-to-light (green) data points, see Fig. 1 at right.

To estimate the drag latency $L$ for a given measurement, we try to find the latency estimate $\hat{L}$ that minimizes the sum of squared residuals between linear regressions fitted to the 'top' and 'bottom' laser-event data separately and the corresponding latency-compensated touch events:

$$\hat{L} = \arg\min_{L} \; J(L) \tag{1}$$

with:

$$J(L) = \sum_{i=1}^{N/2} (y^*_{\text{top}}[i] - \hat{y}_{\text{top}}[i; L])^2 + \sum_{i=1}^{N/2} (y^*_{\text{bottom}}[i] - \hat{y}_{\text{bottom}}[i; L])^2 \tag{2}$$

In the cost function given by Eq. (2), $N$ indicates the total number of laser events, i.e., 'top' and 'bottom' combined. Furthermore, $y^*_{\text{top,bottom}}$ are linear regressions fitted to the horizontal ($x$) and vertical ($y$) event data for 'top' and 'bottom' separately, i.e., $y^*_{\text{top,bottom}} = a_{\text{top,bottom}} x_{\text{top,bottom}} + b_{\text{top,bottom}}$. Finally, $\hat{y}_{\text{top,bottom}}$ indicates the touch location data corrected for a drag latency $L$, which is calculated by interpolating the measured touch coordinates at time $t$ to time $t + L$. To ensure accurate linear regression fits, the horizontal screen position was varied in the measured data, by moving the center of the circular reference path used for the latency measurements (see Fig. 1) left to right across the screen with a constant velocity.

Previous human-in-the-loop experiments with touchscreen input devices performed at our group [4, 5], as well as our measurement setup, were implemented in the Linux-based DUECA real-time simulation environment [18]. Equivalent to these earlier experiments, our current testing software used a constant update rate of 100 Hz, for which the last available touch-event was always used at each time step as the real input.

### B. Apparatus

For the dragging inputs a stylus was used (Wacom Bamboo Alpha, with a diameter $D_{stylus}$ of 70 mm), allowing for more control over the stylus' orientation and contact area [19]. Laser measurements were performed using the OMRON E3X-HD41 2M + E32-T11N laser sensor and a Beckhoff EK1101 and EL1252 EtherCAT module, see Fig. 2(a), whose hardware clock was used to time tag the data. The laser sensor itself has a response time of 55 $\mu$s*, whereas the Beckhoff modules introduce an interface delay of about 1 $\mu$s†. The laser was mounted on the touchscreen as shown in Fig. 2(b). Fig. 2(b) also shows the blue target circle that was displayed during measurements in order to reach the desired input speed.

For reference, the dragging latency as well as the tap latency were also measured using the WALT latency timer‡, see Fig. 2(c). This device was mounted on a spring-loaded pen, to use the WALT's built-in accelerometer to detect the physical time of a tap event (touch or release) and compare it to the registered time of the Linux kernel. The copper strip, shown on the left-hand side of Fig. 2(c), is the conductive element that forces a touch event in the Linux kernel. One hundred taps were performed to get a reliable estimate, of which the median was used as the final measured tap latency.

The computer used for our measurements was a Dell precision T1600 with an Intel Xeon E31245 CPU, running Ubuntu Linux 18.04 with the 4.15 LTS kernel.

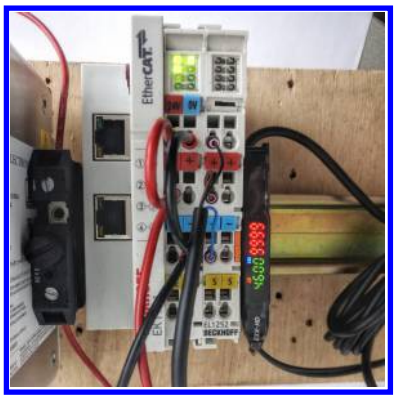
### C. Verification Procedure

Verification of the set-up was performed by comparing the outputs of the different sensors in Fig. 2. While the delay of the WALT latency timer itself is unknown, its hardware clock drift was measured and found to be below 1 ms per

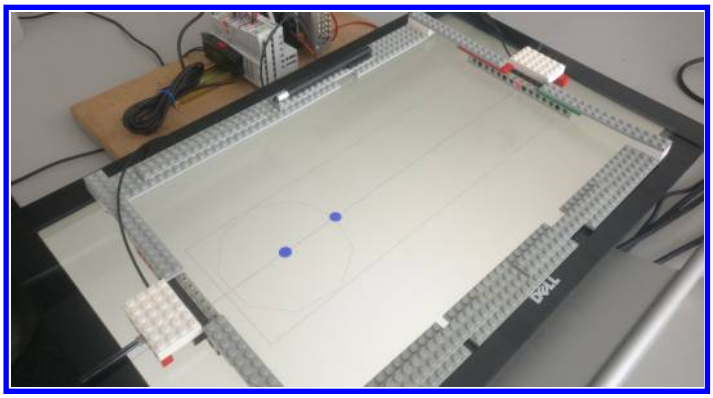
*Set in USH mode, with a threshold of 3900, see the OMRON E3X-HD41 2M + E32-T11N manual.

†Beckhoff EK1101 manual and the Beckhoff EL1252 manual.

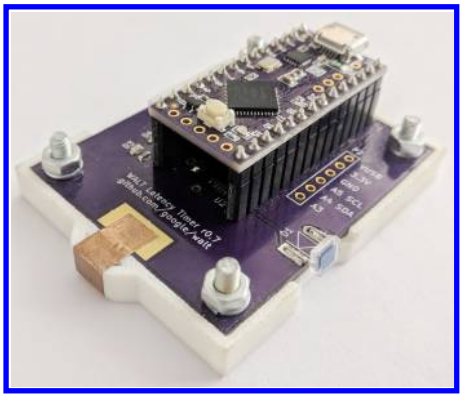
‡https://github.com/google/walt



(a) OMRON laser with EtherCAT interface



(b) Laser measurement set-up



(c) Walt Latency Timer

**Fig. 2 Hardware used for touchscreen drag latency measurements.**

60 s. The code used for the WALT latency timer is available on the WALT's Github page, with the exception of an additional hardware clock synchronization function. The code for the EtherCAT module was custom and developed in-house. The first verification performed consisted of a direct comparison of the touch latency measured at very high input speeds and the WALT tap latency measurements. As for an infinitely fast dragging maneuver the touch event data will not be subjected to jitter-averaging, the drag-latency at high input speeds and the tap latency should be equivalent.

As an additional sanity check, the known diameter of the stylus was compared to the measured distance between the regression lines going through the laser events above and below the laser beam, see Fig. 1. These should be parallel, given that both are defined by the start (light-to-dark) and the ending (dark-to-light) of the occlusion of the laser sensor by the stylus. In reality, measurement noise will force the lines to not be perfectly parallel. Hence, an approximation was obtained by using 5 equidistant points distributed along the 'top' regression line and averaging the perpendicular distances to the 'bottom' regression line using Eq. (3):

$$\hat{D}_{stylus} = \frac{1}{5}\sum_{i=1}^{5} \frac{|b_{\text{bottom}} + a_{\text{bottom}} x_{\text{top}}[i] - y_{\text{top}}[i]|}{\sqrt{1 + a_{\text{bottom}}^2}} \tag{3}$$

In this equation $a_{\text{bottom}}$ and $b_{\text{bottom}}$ are the regression coefficients of the 'bottom' linear regression, while $x_{\text{top}}[i]$ and $y_{\text{top}}[i]$ are the selected points on the 'top' regression line.

### D. Validation Procedure

The measured drag latency as a function of input speed obtained with the approach outlined above was validated using additional data measured with a number of different input patterns relevant for typical human-in-the-loop experiments:

1) A *chirp signal*. For the chirp signal, the frequency of the circular movements as also used for the latency measurements was now increased linearly from 0.05 Hz to 1 Hz (input radius was 20% of the screen's lateral resolution) in 80 s. The horizontal speed component for this chirp signal was set to 15 px/s.
2) A *multi-sine signal*. A tracking task with a multi-sine target signal, see Table 1 and Fig. 3, with a period of 40.96 s and a total measurement time of 45 s was performed on the touchscreen. Also here the additional horizontal speed component was set to 15 px/s.
3) A *step tracking signal*. Finally, a tracking task with a step target signal was performed. Noting that the movement velocity for such tasks is bell-shaped, the target signal was created such that the laser beam is crossed at 5%, 25%, 50%, 75%, 95% of that bell curve, for a step target that is 40% of the maximum longitudinal distance.

**Table 1 Multi-sine signal parameters.**

| Freq. index | Freq., rad/s | Amplitude, - | Phase, rad |
|---|---|---|---|
| 4 | 0.614 | 1.079 | 7.239 |
| 7 | 1.074 | 0.776 | 0.506 |
| 13 | 1.994 | 0.391 | 7.86 |
| 19 | 2.915 | 0.225 | 8.184 |
| 29 | 4.449 | 0.117 | 9.012 |
| 37 | 5.676 | 0.082 | 6.141 |
| 43 | 6.596 | 0.066 | 6.776 |
| 59 | 9.050 | 0.051 | 6.265 |
| 79 | 12.118 | 0.035 | 4.432 |
| 109 | 16.720 | 0.028 | 2.672 |
| 157 | 24.083 | 0.024 | 8.009 |

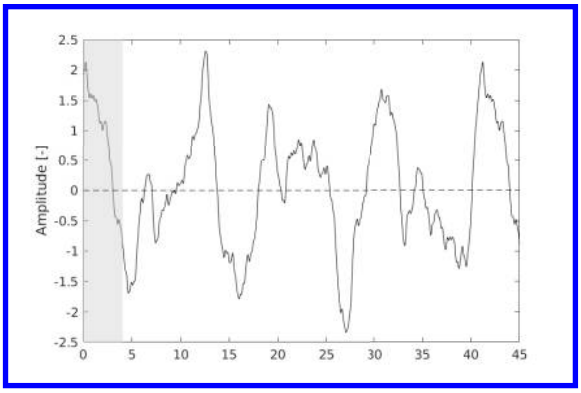



**Fig. 3 Time trace of the multi-sine signal.**

The prediction performance of the drag latency model was quantified using the cost function $J(L)$ of Eq. (2):

$$\Delta J = \left[1 - \frac{\bar{J}(\hat{L})}{\bar{J}(0)}\right] \cdot 100\% \tag{4}$$

In this equation, $\bar{J}(0)$ is the sum of squared residuals of the uncorrected laser events (i.e., $L = 0$ s) for a straight curve through the top and bottom laser events, normalized with the number of laser events (indicated with the bar over $J$). $\Delta J$ is the relative improvement obtained with the laser events shifted with the latency model. Its respective cost is also normalized with the number of laser events and defined as $\bar{J}(\hat{L})$. Hence, if the predicted latency $\hat{L}$ is accurate, $\bar{J}(\hat{L})$ is close to zero and hence $\Delta J$ is 100%, to indicate all latency has been accounted for in the model.

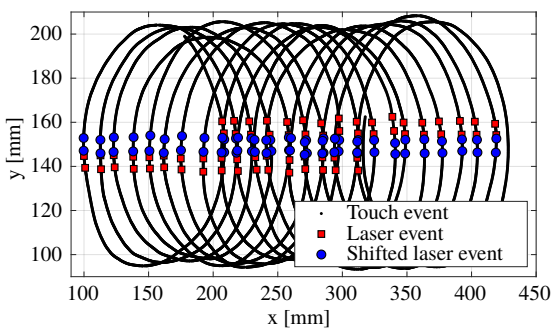
## III. Results

### A. Drag Latency Measurements and Model

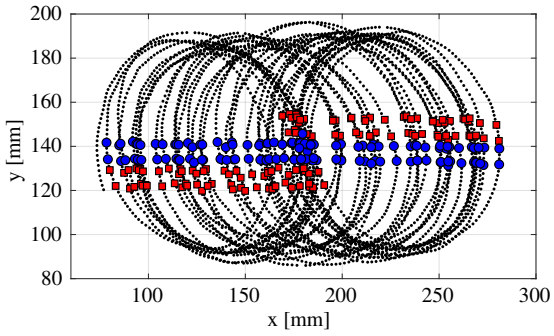
Fig. 4 shows an example of raw drag latency measurement data obtained for Screen 1 (Fig. 4(a)) and Screen 2 (Fig. 4(b)) at different constant angular velocities of the circular drag input signal. Fig. 4 shows the touch sensor data in black, with the raw and latency-corrected laser events indicated with red and blue markers, respectively. As is clear from both sets of raw measurements, a constant time shift can align the raw laser events (red markers) such that the 'top' and 'bottom' laser events are shifted to lie on two straight lines (blue markers). This is the case for both touchscreens. Measurements as shown in Fig. 4 were performed for a wide range of input speeds between 44 and 1000 mm/s. The resulting measured latency characteristics for Screens 1 and 2 are shown in Fig. 5.

Fig. 5 shows that for the tested devices (Screen 1 and 2) the drag latency is indeed strongly dependent on the input speed. Consistent with the outcome of a previous experiment [4], the drag latency of Screen 1 is found to be notably larger than for Screen 2. Overall, the drag latency measurements obtained from the WALT latency timer (red triangles) and the laser set-up (dark blue squares) for Screen 1 are found to be highly consistent. The latency plateau that is observed for lower input speeds for Screen 2 in Fig. 5 (for input speeds below 100 mm/s), shows the use of a jitter filter on the touch input, a design trade-off for touchscreen manufacturers. As is clear from our data, such a plateau was not found with the tested input-speeds for Screen 1.

Based on the measured data shown in Fig. 5, we modeled the trend in the drag latency $L$ with input speed $v$ using an exponential decay-curve model:




(a) Screen 1: Input speed of 70.51 mm/s, latency of 120.2 ms.




(b) Screen 2: Input Speed of 571.26 mm/s, latency of 29.74 ms.

**Fig. 4 Examples of raw latency measurement data for Screens 1 and 2 at different input speeds.**

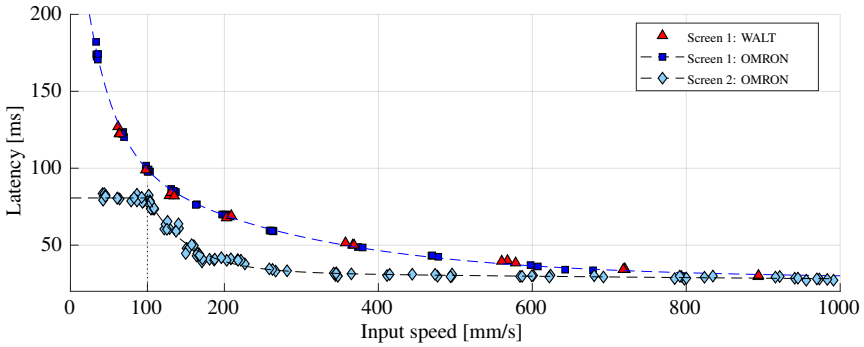



**Fig. 5 Drag latency as a function of input speed for Screens 1 and 2 with fitted models.**

$$L(v) = a_1 e^{-a_2(v-a_3)} + a_4 e^{-a_5(v-a_6)} + a_7 \tag{5}$$

In this equation, $a_i$ are the drag-latency model's coefficients, for which the estimated values corresponding to the data in Fig. 5 are listed in Table 2. In fitting the measured data, the $a_3$ and $a_6$ coefficients were found to be redundant for describing our data and not estimated, but they are still included in Eq. (5) for generality. The true input speed ($v$) was estimated from touch-event data using the average registered input speed at the time of laser events. Before estimating $v$, the touch input data was interpolated for Loss of Contact (LOC) events and filtered using a Savitsky-Golay filter [20] of order 4 and a window length of 61 samples.

**Table 2 Drag-latency model coefficients and normalized RMS error for the data of Fig. 5.**

| | $a_1$ | $a_2$ | $a_4$ | $a_5$ | $a_7$ | RMS($L - \hat{L}$) |
|---|---|---|---|---|---|---|
| Screen 1 | 90.53 | 4.00e-3 | 182.9 | 2.90e-2 | 28.49 | 1.43 ms |
| Screen 2 | 9.90 | 7.92e-2 | 313.7 | 1.88e-2 | 23.65 | 2.19 ms |
| Screen 2 (corrected) | 9.57 | 3.30e-3 | 440.7 | 1.98e-2 | 28.22 | 2.24 ms |

Fig. 5 shows that the exponential drag-latency model can accurately describe the measured data for both screens. This is also supported by the low root mean square errors of the drag-latency model with respect to the data, i.e., RMS($L - \hat{L}$), as listed in Table 2. For Screen 2, Table 2 also shows a corrected estimation result, with a slightly higher RMS error, that was used as the final model fit as the $a_7$ coefficient (asymptote) was underestimated compared to the measured latency at input speeds around 1000 mm/s (see also Section III.B).

### B. Verification: Correspondence of Tap and Drag Latency

As tapping on a touchscreen is assumed to be processed similarly as an infinitely fast dragging maneuver, it is expected that the asymptote of the drag-latency characteristics for both screens as shown in Fig. 5 is equivalent to the

tap latency measured with the WALT timer, see Section II.C. The asymptotic value of the drag-latency characteristic is also modeled as the $a_7$ coefficient of the drag-latency model, see Table 2. Fig. 6 shows the measured tap latency data obtained using the WALT timer, both for 'touch' and 'release' events. For Screen 1 the $a_7$ parameter (28.49 ms) is indeed found to be approximately equal to the median of the release tap events (28.75 ms). For Screen 2, the median release tap latency is found to be 28.14 ms, while the asymptotic drag latency was 23.65 ms, see Table 2. However, the measured latency values around an input speed of 1000 mm/s were in fact found to match the tap latency: 28.22 ms. Hence, a corrected estimate of the drag-latency model was obtained for Screen 2 by forcing the asymptote to be equal to tap latency, see Table 2. Overall, the laser-sensor measurements of the drag latency were thus verified by their correspondence with the WALT timer tap latency measurements.

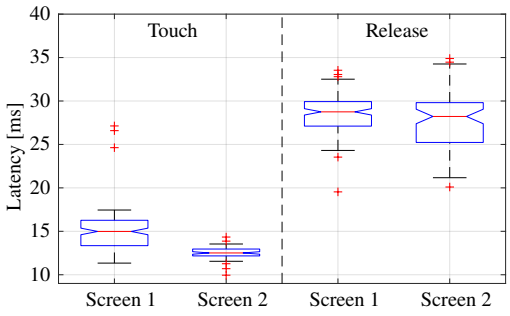



**Fig. 6 Tap latency measured by the WALT latency timer ($N$=100).**

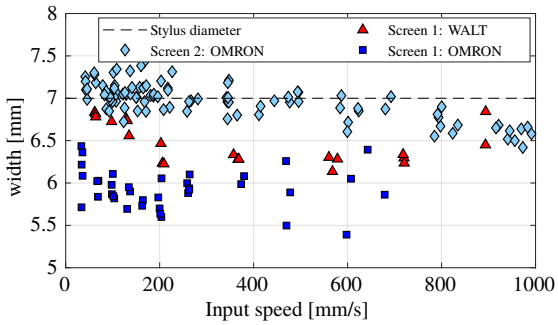



**Fig. 7 Stylus width estimated from the vertical offset between the 'top' and 'bottom' regression lines.**

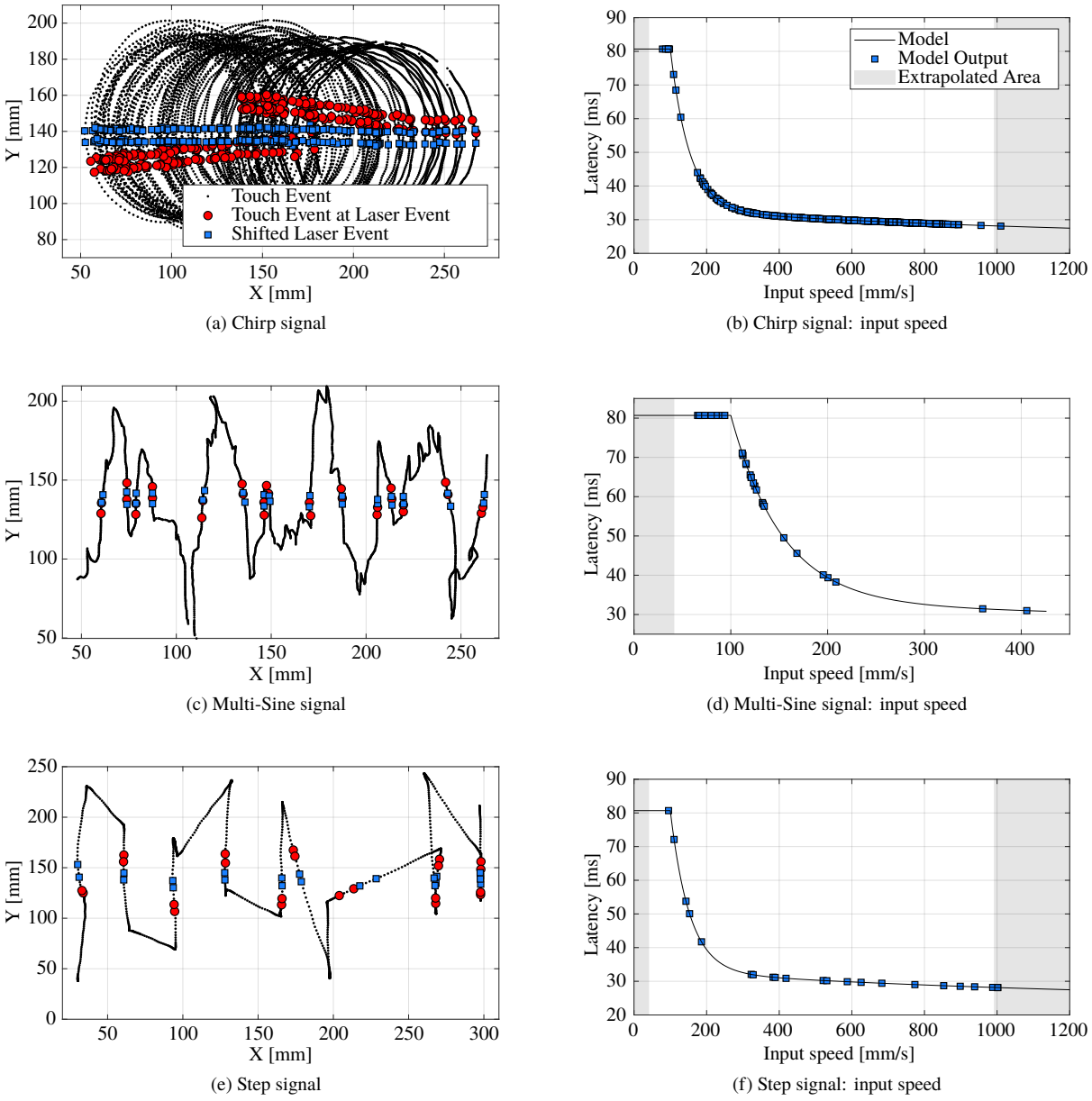
### C. Verification: Stylus Width

As an additional verification of the measurement setup, we compared the known stylus width with a 'measured' stylus width estimated from the vertical offset between the 'top' and 'bottom' regression lines that were fitted through the laser-event data, see Fig. 1. Fig. 7 shows the estimated stylus width as obtained from the data from both screens using Eq. (3). Fig. 7 shows that while the stylus width estimates for Screen 2 (light blue diamonds) accurately match the true $D_{stylus}$ of 7 mm (dashed black line), the stylus width estimates obtained from Screen 1 are consistently around 1 mm too low. This difference is attributed to the more consistent timing of the touch event data received from Screen 2, which results in more accurate fitted regression lines for the 'top' and 'bottom' event data. Despite the explainable mismatch in the results for Screen 2, overall the estimated stylus width data in Fig. 7 adds to the verification of our approach.

### D. Validation: Latency Model Application

The relation between the input speed $v$ and the incurred drag latency $L$ measured for Screens 1 and 2, as modeled using the exponential decay-curve model of Eq. (5), was validated using data from three different touch input sequences with varying input speed: a chirp signal, a multi-sine signal, and a step signal. Example data for Screen 2 with three signals is shown in Fig. 8. The (circular) chirp signal was similar to the screen movements used for estimating the latency characteristic, only with a linearly increasing input speed. Matching Fig. 4, Figures 8(a), (c), and (e) show the uncorrected laser-event data in red, while the corrected data (here corrected with the previously estimated drag-latency model) is shown with blue markers. Figures 8(b), (d), and (f) show the distribution of input speeds at the laser crossings for all three cases, superimposed on the drag-latency characteristic of Screen 2.

Fig. 8 indeed shows that correction of the laser-event data for these validation cases indeed results in corrected event positions (blue markers) that line up and have approximately the same vertical screen positions. To further quantify this, Eq. (4) was used to calculate a performance improvement in the alignment of the laser events between the raw and corrected data. The $\Delta J$ performance metric, averaged over 3 repeated measurements for each case, is presented in Table 3. For reference, also the cost for the uncorrected data ($J(0)$), normalized with the number of laser events ($N$), is listed for comparison. Fig. 8 and Table 3 show that the exponential drag-latency model of Eq. (5) with the estimated model parameters of Table 2 indeed effectively corrects for the (varying) input-speed related drag-latency for all three input patterns. On average, the linear regression fit to the corrected data is at least 90% more accurate than the




(a) Chirp signal

(b) Chirp signal: input speed

(c) Multi-Sine signal

(d) Multi-Sine signal: input speed

(e) Step signal

(f) Step signal: input speed

**Fig. 8 Example validation tests as performed for Screen 2.**

corresponding fit to the uncorrected data for both screens. Hence, the derived drag-latency models accurately capture the screens' input speed-dependent latency.

## IV. Conclusions

This paper proposed a novel method for measuring the drag latency of touchscreens as an essential addition to human-in-the-loop experiments making use of such devices. The methodology was tested on two touchscreens, from different vendors, that were used for previous human-in-the-loop experiments. Overall, the obtained results show that a wide variation in drag latency (28-200 ms) can be expected between different input speeds (i.e., tasks), as well as touchscreen hardware (i.e., Screens 1 and 2). Following an exponential decay-curve, drag latency is found to be much larger at low input speeds than at high input velocities. This is consistent how slow-moving touches are commonly

**Table 3 Model improvements for the three validation input patterns.**

| | Screen 1 | | Screen 2 | |
|---|---|---|---|---|
| | $J(0)/N$ [mm$^2$] | $\Delta J$ [%] | $J(0)/N$ [mm$^2$] | $\Delta J$ [%] |
| Chirp | 206.71 | 99.51 | 82.94 | 99.36 |
| Multi-sine | 71.09 | 88.02 | 34.34 | 96.39 |
| Step | 334.07 | 96.73 | 314.55 | 95.42 |

chosen to require better accuracy, and thus more averaging in touch sensor data processing, than faster movements. For both screens, the drag-latency characteristic measured with our proposed approach was modeled and the models were found capable of accurate compensation (>90% on average) for the drag latency in three touch input tasks with varying input speeds representative for the tasks typically performed human-in-the-loop experiments. Overall, these findings help explain observed task performance discrepancies between different touchscreens as encountered in previous experiments, e.g., [4]. We conclude that for valid human-in-the-loop experimentation with time-based task paradigms these findings imply that detailed knowledge of the touch hardware's drag-latency characteristics is essential for ensuring valid and repeatable human-in-the-loop experiments.



## References

[1] Dodd, S. R., Lancaster, J., Grothe, S., DeMers, B., Rogers, B., and Miranda, A., "Touch on the flight deck: The impact of display location, size, touch technology & turbulence on pilot performance," *Proceedings of the IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC)*, 2014. https://doi.org/10.1109/DASC.2014.6979428.

[2] Rouwhorst, W., Verhoeven, R., Suijkerbuijk, M., Bos, T., Maij, A., Vermaat, M., and Arents, R., "Use of touch screen display applications for aircraft flight control," *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, IEEE, 2017, pp. 1–10. https://doi.org/10.1109/DASC.2017.8102060.

[3] Cockburn, A., Gutwin, C., Palanque, P., Deleris, Y., Trask, C., Coveney, A., Yung, M., and MacLean, K., "Turbulent Touch: Touchscreen Input for Cockpit Flight Displays," *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, Denver (CO)*, 2017, pp. 6742–6753. https://doi.org/10.1145/3025453.3025584.

[4] Mobertz, X. R. I., Pool, D. M., Van Paassen, M. M., and Mulder, M., "A Cybernetic Analysis of Biodynamic Effects in Touchscreen Operation in Turbulence," *Proceedings of the AIAA Modeling and Simulation Technologies Conference, Kissimmee (FL)*, 2018. https://doi.org/10.2514/6.2018-0115.

[5] Van Zon, N. C. M., Borst, C., Pool, D. M., and Van Paassen, M. M., "Touchscreens for Aircraft Navigation Tasks: Comparing Accuracy and Throughput of Three Flight Deck Interfaces Using Fitts' Law," *Human Factors*, Vol. 62, No. 6, 2020, pp. 897–908. https://doi.org/10.1177/0018720819862146.

[6] Khoshnewiszadeh, A., and Pool, D. M., "Mitigation of Biodynamic Feedthrough for Touchscreens on the Flight Deck," *International Journal for Human-Computer Interaction*, Under review. Special Issue on "Modern Flight Deck".

[7] Padre, J., "Understanding touch responsiveness; Touchscreen technology series 2," https://developer.sonymobile.com/2014/07/02/understanding-touch-responsiveness-touchscreen-technology-series-2, 2014. Accessed June 11, 2018.

[8] Stroosma, O., Mulder, M., Postema, F. N., and Van Paassen, M. M., "Measuring Time Delays in Simulator Displays," *AIAA Modeling and Simulation Technologies Conference and Exhibit, AIAA 2007-6562*, Vol. 6562, No. August, 2007, pp. 1–9. https://doi.org/doi:10.2514/6.2007-6562.

[9] Pavlovych, A., and Stuerzlinger, W., "The tradeoff between spatial jitter and latency in pointing tasks," *Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems - EICS '09*, 2009, p. 187. https://doi.org/10.1145/1570433.1570469, URL http://portal.acm.org/citation.cfm?doid=1570433.1570469.

[10] Pavlovych, A., and Gutwin, C., "Assessing Target Acquisition and Tracking Performance for Complex Moving Targets in the Presence of Latency and Jitter," *Proceedings of Graphics Interface 2012*, 2012, pp. 109–116.

[11] 3M, "3M Touch Systems, Touch Technology Brief," http://multimedia.3m.com/mws/media/788463O/tech-brief-projected-capacitive-technology.pdf, 2013.

[12] Ng, A., Lepinski, J., Wigdor, D., Sanders, S., and Dietz, P., "Designing for low-latency direct-touch input," *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*, ACM, ACM Press, New York, New York, USA, 2012, p. 453. https://doi.org/10.1145/2380116.2380174, URL http://dl.acm.org/citation.cfm?doid=2380116.2380174.

[13] Kaaresoja, T., and Brewster, S., "Feedback is... late," *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction on - ICMI-MLMI '10*, ACM Press, New York, New York, USA, 2010, p. 1. https://doi.org/10.1145/1891903.1891907, URL http://portal.acm.org/citation.cfm?doid=1891903.1891907.

[14] Jota, R., Ng, A., Dietz, P., and Wigdor, D., "How fast is fast enough?: a study of the effects of latency in direct-touch pointing tasks," *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2013, pp. 2291–2300. https://doi.org/10.1145/2470654.2481317, URL http://dl.acm.org/citation.cfm?doid=2470654.2481317.

[15] MacKenzie, I. S., and Ware, C., "Lag as a determinant of human performance in interactive systems," *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '93*, 1993, pp. 488–493. https://doi.org/10.1145/169059.169431, URL http://portal.acm.org/citation.cfm?doid=169059.169431.

[16] Gordon, J., Ghilardi, M. F., Cooper, S. E., and Ghez, C., "Accuracy of planar reaching movements," *Experimental Brain Research*, Vol. 99, No. 1, 1994, pp. 112–130. https://doi.org/10.1007/BF00241416, URL http://link.springer.com/10.1007/BF00241416.

[17] Soukoreff, R. W., and MacKenzie, I. S., "Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI," *International journal of human-computer studies*, Vol. 61, No. 6, 2004, pp. 751–789. https://doi.org/10.1016/j.ijhcs.2004.09.001.

[18] Van Paassen, M. M., Stroosma, O., and Delatour, J., "DUECA - Data-Driven Activation in Distributed Real-Time Computation," *Proceedings of the AIAA Modeling and Simulation Technologies Conference and Exhibit, Denver (CO)*, 2000.

[19] Forlines, C., Wigdor, D., Shen, C., and Balakrishnan, R., "Direct-touch vs. mouse input for tabletop displays," *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07*, ACM Press, New York, New York, USA, 2007, p. 647. https://doi.org/10.1145/1240624.1240726, URL http://portal.acm.org/citation.cfm?doid=1240624.1240726.

[20] Schafer, R. W., "What is a Savitzky-Golay filter?" *IEEE Signal Processing Magazine*, Vol. 28, No. 4, 2011, pp. 111–117. https://doi.org/10.1109/MSP.2011.941097.