Effects of Latency Jitter and Dropouts in Pointing Tasks

ABSTRACT
Interactive computing systems frequently use pointing as an input modality, while also supporting other forms of input. We focus on pointing and investigate the effects of variations, i.e., jitter, in the input device latency, as well as dropouts on 2D pointing speed and accuracy. First, we characterize the latency, latency jitter, and dropouts in several common input technologies. Then we present an experiment, based on ISO 9241-9, where we systematically explore combinations of dropouts, latency, and latency jitter on a desktop mouse to measure how these factors affect human performance. The results indicate that latency and dropouts have a strong effect on human performance, while moderate amounts of latency jitter do not change performance in a statistically significant way in most cases.

The findings are useful for the design of pointing technologies for interactive systems. Our results permit prediction of the impact of various detrimental factors on performance, which provides guidelines for dealing with latency, latency jitter, and dropouts.

Author Keywords
Latency, jitter, Fitts’ law, pointing.

ACM Classification Keywords
H.5.2. Information Interfaces and Presentation: User Interfaces – Theory and methods; evaluation/methodology.

INTRODUCTION
While the mouse is the most common pointing device in human-computer interaction, there is a large variety of other devices that have appeared over the years. One of the most recent additions is the Nintendo Wii Remote. Although virtually all of these devices can be used in interactive computing systems, most of them exhibit significantly more latency than the mouse. Moreover, some implementation details, environmental conditions, as well as technological choices can impact variations in latency, as well as dropouts in the report of positional information. The best example here is the widespread employment of wireless interfaces, which may lead to interference and hence to transmission problems up to intermittent connection drops. Consider that mice and Nintendo’s Wii Remotes may use the same Bluetooth 2.5 GHz unlicensed radio band, which is also occupied by cordless telephone handsets, wireless local area networking technologies, and microwave ovens.

Latency, or lag, is the delay in device position updates [9]. Latency and spatial jitter have been previously demonstrated to significantly impact human performance in both 2D and 3D tasks [17, 21, 22, 25]. However, in popular computing systems, it is technically hard to guarantee constant latencies at frequencies that are necessary for high-performance pointing, usually 100 Hz or better. Hence, all systems have to deal with varying degrees of latency variations, better known as latency jitter or just jitter. This is a phenomenon that has been studied in detail in the area of computer networking [14]. There, the latency variations are typically caused by network load variations and by buffering of data packets on the network nodes. The work on latency jitter concentrated on optimizing network performance, but no systematic studies were done to investigate the immediate effect of this on human-computer interaction. Armitage et al. [3] conducted a study of jitter in network gaming, but the authors were not able to determine the effects of latency jitter on performance, due to high correlation between latency and latency jitter in their collected data. Recent interest in remote application use (application as a service, [23]), as well as a renewed interest in interactive network gaming [20] further highlights the need for systematic study of this phenomenon.

Finally, all pointing devices are based on some sensor technology. These technologies are affected to varying degrees in the reliability of position tracking, ranging from very high (e.g., high-quality mouse on a “good” surface), to good (e.g., video-based object tracking under lab conditions), to mediocre (e.g., optical mouse on a glossy surface), to unusable (e.g. video-based tracking in a constantly changing environment, including lighting changes). Any failure of the sensing gives rise to dropouts in the sequence of position reports. Dropouts can also occur
when the motion being sensed exceeds the devices’ capabilities, e.g., a maximum linear velocity for a mouse, or the ability of the network to transmit high frequency signals on time. Most of the current device development focuses on disparities in latency and spatial jitter of the employed devices [25]. Furthermore, dropout behaviour can easily be simulated in software, to match that of any other device. Ultimately, we believe that many classes of devices can be emulated with a mouse, if one artificially adds latency, latency jitter, as well as dropouts to an “ideal” mouse to simulate the limitations of a particular technology.

**Pointing Systems, Lag and Jitter**

Latency is the time from when the device is physically moved to the time the corresponding update appears on the screen. For technical reasons, it is hard to avoid latency. And it is known that latency adversely affects human performance in both 2D pointing [17] and 3D pointing [6, 26].

Spatial jitter is caused either by hand tremor or noise in the device signal or both. One way to observe noise is to immobilize a tracking device while observing the reported positions. However, some devices also exhibit additional noise during movements. Hand jitter only exacerbates this problem, especially in devices used in free-space.

Temporal jitter, or latency jitter, refers to changes in lag with respect to time. Ellis et al. [7] report that people can detect very small fluctuations in lag, likely as low as 16 ms. Hence, when examining system lag, one must also ensure that latency jitter is minimized, or at least known. However, it is unknown if this ability to detect latency changes translates into drops in human performance.

For 3D tracking devices, Foxlin [9] provided a thorough overview of the various tracking technologies. Although the author argues that one should choose a specific tracking technology based on technological needs [9], most tracking technologies have shortcomings that may also affect human performance.

**Fitts’ Law**

Fitts’ law [8] is a model for repeated rapid, aimed movements. It is expressed as:

\[ MT = a + b \cdot \log_2 \left( \frac{A}{W} + 1 \right) \tag{1} \]

where \( MT \) is the movement time, \( A \) the amplitude of the movement (i.e., the distance between targets), and \( W \) is the width of a target. The log term is called the Index of Difficulty (ID), and commonly assigned a unit of bits:

\[ MT = a + b \cdot ID \tag{2} \]

The coefficients \( a \) and \( b \) are usually determined empirically for a given device and interaction style, such as a stylus on a tablet, a finger on an interactive tabletop, etc.

The interpretation of the equation is that movement tasks are more difficult when the targets are smaller or farther away. Fitts’ law has been used to characterize the performance of pointing devices and is one of the main

**BACKGROUND**

This section briefly discusses relevant work in object manipulation, tracking and pointing technology, and Fitts’ law. Many different pointing technologies can be utilized in interactive systems today. In addition to the computer mouse, the major alternatives are (with exemplary references):

- Touch pads (used in laptops)
- Touch screen technologies [2], [12])
- Laser pointers (used for distant pointing [19])
- Video-based marker tracking (distant pointing, [4])
- Video-based hand tracking [10]
- Accelerometer-enhanced devices (gyro-mouse, gaming devices, tilt-based interaction, [16])
- General 3D tracking systems, usually via optical, ultrasonic, electromagnetic, or inertial means.

Our current study uses the mouse as an input device. The first reason for that is that the mouse is has much lower levels of latency, jitter of any form, or dropouts compared to most other technologies. Another is that the mouse is a well-studied and established device. Also, previous work comparing mouse-based and tracker-based manipulation techniques found that the differences between mice and trackers can be explained to a large degree by the

We present two empirical studies that systematically investigate the effects of dropouts and latency jitter on human performance. The studies employ Fitts’ law, a well-established model of pointing device performance. In our experiments, we used a mouse as an exemplary low-latency, low-jitter device, and artificially added latency and latency jitter to it, to match the range of latencies and jitter present in other commonly used devices, as well as in computer networks. We also varied the number and the percentage of samples that the system was omitting (“dropping”) during the experiment. The main goal was to determine, all else being equal, the effects of dropouts and latency jitter on device performance at varying mean latencies. Seen from another viewpoint, we are investigating, which has a stronger impact on human performance: latency, latency jitter, or dropouts? As one can often trade some latency for a decrease in latency jitter, typically through time-domain filtering, and extrapolate the missing and delayed samples, knowing the interrelationships between the factors allows a designer to make an informed decision in choosing an appropriate filter and its parameters.
components of standardized evaluations in accordance with ISO 9241-9 [13]. Indeed, if the movement time and \( ID \) are known, then their ratio defines the throughput \( BW \) of the input device in bits per second [bps]:

\[
BW = \frac{ID}{MT}
\]

(3)

**Effective Width and Effective Distance**

During a traditional Fitts’ task, participants are asked to click on targets of various sizes, spaced at various distances. Usually they hit larger targets with fewer misses and relatively closer to their centers. Smaller targets are missed more often and clicks may occur farther away from their centers. Thus, and especially for imperfect input devices, it is beneficial to take this variation into account. As an illustration, Figure 1 shows a typical distribution of hits when a task is performed repeatedly with a mouse.

MacKenzie argues for using a sub-range of the hit data, corresponding to about 96%, as the effective width of the target [15]. This range corresponds to approximately 4.133 standard deviations of the observed coordinates of hits, relative to the intended target center:

\[
W_e = 4.133 \cdot \sigma
\]

(4)

MacKenzie points out that this practice corresponds better to the task that the user actually performed, compared to the ideal Fitts task. Moreover, this compound measure of performance and accuracy also eliminates the need for a separate error analysis across input device technologies.

![Figure 1. Distribution of clicks on a circular target [21].](image)

To calculate the effective parameters, we projected the actual movement vector onto the intended vector and computed the difference of the vector lengths as the deviation from the intended center. This allows us to later compute the effective width using equation (4) above. A similar approach is used for distance: the actual movement distances are measured and then averaged over all repetitions, thus forming the effective distance. Figure 2 illustrates both notions. Finally, both effective distance and effective width, in combination with movement time, are used to determine the effective throughput of a device, a measure that takes not only the performance but also the accuracy of target acquisitions into account.

![Figure 2. Illustration of Effective Width and Effective Distance. These are averaged over multiple movements [21].](image)

We use these effective measures in place of the true target widths and amplitudes to seamlessly incorporate the potential effect of differing participant strategies, which favour speed or accuracy [15], as well as differences in technology. In essence, the approach treats more accurate clicks, i.e., clicks closer to the centre of the targets, as clicks on smaller targets, while the clicks outside of the intended targets are treated as “successful” clicks on larger virtual targets. This independence from user strategies and technologies is one of the reasons why this measure is recommended by ISO 9241-9 for pointing devices [13].

**Characterizing System Latency and Time Jitter**

Before performing experiments, we quantified the end-to-end system latency of our setup to establish a baseline condition.

The goal in building and tuning our setup was to minimize the system’s baseline latency, as this enables us to simulate other systems via software. Consequently, we used a Microsoft wheel optical mouse on an office table surface, as such mice are generally very accurate, reliable, and smooth in sensing motion. Moreover, the mouse reports updates at 125 Hz with minimal latency. The friction induced by the surface also effectively dampens any exterior oscillations, and we were not able to observe noticeable spatial jitter when the mouse was stationary or moving in straight line. We used a 21” CRT display at 1024 × 768 pixels and 120 Hz as output device, as such monitors are known for their low latencies and predictability. We also employed

---

1 In the absence of jitter and latency and assuming accurate human pointing, throughput generally does not depend on \( ID \). This makes it a good metric for comparisons.
single-buffering for screen updates on the output side, again to minimize latency.

We understand that most systems today use double-buffering for output, but the effect of this on latency is well understood in the area of computer graphics and virtual reality. Also, common LCD displays have update rates of only 60 Hz and may exhibit lags of 40 ms [21]. If a DLP projector is used, latencies as high as 100 ms may be encountered, and many computer games have significant delays, with 80–150 ms being most common [5]. But all such effects can easily be modeled based on the known relationships between the latency and performance, see e.g., [21]. The discussed range of latencies was used to choose the investigated set of latencies in our experiments.

**Characterizing Latency**
A variation of Mine’s method was used to characterize the lag of the mouse [18].

**Equipment Setup & Procedure**
The mouse was moved along the top bezel of the CRT display. The area where the mouse moved was covered with a textured material to ensure reliable tracking. When the mouse was moved, the cursor on the screen moved correspondingly. For consistency, we used the same software that we subsequently employ for our experiments, with all factors, such as additional latency, set to zero. To measure the latency, a video camera simultaneously filmed the motion of both the mouse and the cursor at a frame rate of 60 Hz in progressive scan mode. The mouse was repeatedly moved by hand sideways along the top edge of the display bezel at a rate of about 1 Hz. The end-to-end latency of the system then equals the average differences in the frame times of their corresponding phases of motion.

**Analysis and Results**
Approximately two minutes of video were recorded with the digital camera. This video was analyzed manually after the experiment to derive the end-to-end latency. Peaks of mouse and cursor movement were examined. When the mouse reached the peak position of its movement in one direction or the other, the frame number and its time were noted. When it began to move back the other way, the mouse cursor on the screen would move back as well, but after a short delay due to system latency. These delays were also recorded. Because the camera was only recording at 60 Hz, we averaged a total of 10 measurements to reduce potential sampling artefacts. Ultimately, the average delay of the mouse cursor motion relative to motion of the mouse was 8 ± 2.8 ms at the center of the screen.

**Latency Jitter and Dropouts**
We also measured latency jitter, i.e., the amount of change in latency from one point in time to another. To measure this, we looked at the mouse update intervals. Our mouse reported updates at 125 Hz, a value typical for most USB mice. A histogram of these times showed that more than 99.5% of the updates happened within 8–11 ms of the previous sample. Practically all of the remaining samples followed within 5–8 ms. We never observed a dropout. Consequently, we do not believe latency jitter to be an issue for the baseline conditions of our experiments, considering that latency variations this small are unlikely to be perceptible [7].

**Spatial Jitter**
Although the optical sensing method employed by the mouse may generate some jitter, this appears to be filtered in the mouse hardware. While the technical details in each specific implementation may differ, typical optical mice sensors are in essence low-resolution miniature video cameras taking images at a rate of several thousand per second [1]. Since a desktop pointing device only requires about a hundred updates per second, the 10:1 or greater excess of frames is apparently used to smooth the device reports via averaging or some other filtering technique.

Likewise, hand jitter, or hand tremor, does not appear to be an issue in our experiments, as resting the mouse on a physical surface largely eliminates tremor. Any oscillation depends on friction, as well as mass, rigidity, and external disturbances. Friction dampens, or reduces, the magnitude of such oscillations and hence also reduces hand tremor. Based on our measurements and the fact that our participants were young, we assume the input had no significant jitter of either kind.

**EXPERIMENT 1**
This experiment used the procedure according to ISO 9241-9 to compare effective throughputs under various magnitudes of latency, time jitter, and dropouts.

**Participants**
Twelve students from the local university participated in the experiment, with ages ranging from 22 to 34 (mean 28). Eight were male. All were right handed, or otherwise used the mouse with their right hand. The study lasted about 40–50 minutes.

**Apparatus**
The computer was an Intel Pentium based desktop with 1 GB of RAM. The software, written in C#, implemented a standard Fitts’ 2D task [13], see Figure 3. The application presented 13 targets in a circle. Upon clicking the first highlighted target (the top one) the timer would start and the opposite (bottom-left) target would be highlighted, directing the participant to select it. The next target was again on the opposite side, to the immediate right of the initial target, and so on until all targets were selected. The
software automatically logged target sizes, distances between targets, the times to click between targets, misses, and screen coordinates of click events. It also performed the effective width calculation as described above.

**Procedure**

After signing informed consent forms, participants were seated in front of the computer display at a distance of about 0.6 m. Participants were given a brief introduction to the system and were allowed to try the system and find the most comfortable seating position. After that, they were directed to proceed with the task, in which they were instructed to click on the highlighted targets as quickly and accurately as possible.

**Design**

The experiment was within subjects, and the order in which the various combinations of the factors were presented was randomized (without replacement), to compensate for asymmetric transfer of learning effects.

The experiment had four independent variables in a \((1 \times 1 + 1 \times 2 + 1 \times 3 + 1 \times 4) \times (3 \times 3 + 1) = 10 \times 10\) arrangement, for a total of 100 combinations:

- **Latency** (constant component): 10*, 40, 100, and 160 ms;
- **Latency jitter** (normally distributed, in addition to the constant value above):
  - \(\sigma = 0^*\) ms for 10 ms latency,
  - \(\sigma = 0, \pm 20\) ms for 40 ms latency,
  - \(\sigma = 0, \pm 20, \pm 40\) ms for 100 ms latency,
  - \(\sigma = 0, \pm 20, \pm 40, \pm 60\) ms for 160 ms latency;
- **Dropout duration**: 0*, 125, 250, 500 ms;
- **Intervals between dropouts**: 0*, 500, 1000, 2000 ms (modeled as a Poisson process with such arrival intervals).

In the above list, * denotes the baseline condition, i.e., minimum latency, no latency jitter, and no dropouts. We rounded the value of the real system latency to 10 ms for convenience.

Six target indices of difficulty (ID), ranging evenly from 2.44 to 5.76 bps, were randomly matched with each of the 100 combinations above.

The dependent variable was effective device throughput (in bits per second), calculated as described earlier.

Each participant completed 100 “rounds” with different latencies, latency jitters, dropout durations, and dropout intervals. As it is not possible to meaningfully measure the click time for the first target, there were only 12 clicks recorded per round. Given that there were 12 participants, this yields a total of 100\(\times\)12\(\times\)12 = 14,400 trials.

The range of latencies considered covers the extent of lags observed in various devices, including the latency of a wireless controller for a gaming console used on a projection screen. The range of IDs covers the span encountered in typical desktop tasks, as well as in other forms of interactions. The highest ID, represented by a target width of 12 pixels, is about the size of a “window close” button in common window managers. Acquiring targets of smaller sizes was observed to be too fatiguing during a small pilot experiment, and hence we decided to restrict the design to values that avoided excessive participant frustration. As the task is highly repetitive, we choose the total number of combinations to keep the total participation time to well less than one hour to keep the fatigue effect as small as possible.

**Results**

Results were analyzed using repeated measures ANOVA. There were significant main effects for all of the independent factors. The effective throughput was computed according to equation (3), using the effective width and amplitude parameters, derived from each set of 12 measurements for every experimental condition.

**Latency**

The effect of latency on throughput was significant, \(F_{3,33} = 200.43, \ p < .0001\). The interaction between the latency and dropout duration was also significant, \(F_{9,99} = 11.59, \ p < .0001\). No other significant interactions for latency were observed. Figure 4 shows a graph of the results. The throughput values in the absence of dropouts are consistent with those reported in [21].
Latency Jitter
The effect of latency jitter on throughput was significant at a latency of 100 ms, $F_{2,22} = 4.81$, $p < .05$, and was approaching significance at latencies of 40 and 160 ms, $F_{1,11} = 4.25$, $p = 0.063$ and $F_{3,33} = 2.70$, $p = 0.061$ respectively. No secondary interactions were observed. Figure 5 illustrates the behavior at 100 ms base latency.

Dropout Duration
The effect of dropout duration on the throughput was significant, $F_{3,33} = 348.66$, $p < .0001$. See Figure 6 for details. The interaction between latency and dropout duration was already noted above.

Interval between Dropouts
The effect of interval between the dropouts on the throughput was significant, $F_{2,22} = 159.00$, $p < .0001$. There was a significant interaction between the dropout interval and duration, $F_{4,44} = 53.44$, $p < .0001$, see Figure 7.

Dropout Percentage
All 9+1 combinations of dropout intervals and durations can also be reformulated as a single Dropout Percentage metric, corresponding to 0, 6, 12, 22, 39, and 63% unreliability. Seen differently, in the last condition, only 37% of the samples were utilized to update the cursor position.

The effect of dropout percentage on the throughput was significant, $F_{5,55} = 293.01$, $p < .0001$, see Figure 8. All points on this graph represent statistically distinct values, except for the pair of 6% and 12%, whose distributions overlap, $F_{1,11} = 1.89$, $p = 0.20$. There was a significant interaction between dropout percentage and latency, $F_{15,165} = 6.74$, $p < .0001$. 

Figure 4. Throughput for varying levels of latency and dropout durations.

Figure 5. Throughput for varying levels of latency jitter at 100 ms base latency. Error bars represent standard error.

Figure 6. Throughput for varying levels of dropout duration.

Figure 7. Throughput for varying levels of dropout intervals. Error bars represent standard error.
DISCUSSION
The throughput of the baseline mouse condition (no added lag, no jitter, no dropouts) is similar to that reported in previous work [24] and we take this as validation of our experimental design. The drop in performance with increased latency is also mirroring the data in previous work [21], which indicates that the latency measurements at the beginning of our user study are accurate.

Weak effect of Latency Jitter
One of the surprising findings was that latency jitter, that is, variations of latency with time, had little effect on performance, resulting in the worst case in an 8.5% drop in performance at 100 ms base latency and jitter with $\sigma \approx 40$ ms. Compared to the dramatic drops with increasing latency or dropouts, see Figure 4, such a small drop is likely to be of little practical significance.

However, we emphasize that we used a normal distribution for the latency jitter component. Many physical phenomena, like system noise, follow a normal distribution, and normal distributions are among the best studied. However, real network latencies are usually described by heavily skewed distributions, such as Pareto or Weibull distributions [27]. In a small pilot experiment we also tested a uniform distribution for latency jitter. The performance of such distributions seems to be strongly determined by the peak latencies, but it is an atypical distribution overall. Hence, we can only point out that the above statements are strongly dependent on the type of latency distribution.

Dropouts
Both dropout durations and the intervals between them noticeably affect performance in a negative way. Both can be characterized by an almost linear dependence, see Figure 4 and Figure 8.

Looking at Figure 4 and Figure 6, it can be seen that moderately long dropouts of 250 ms (12, 22 and 39%, when combined with the durations) have a similar detrimental effect on throughput as the latency of just over 100 ms in the absence of dropouts. Considering that such latency is rather common in some contexts, such as online gaming and use of DLP-based display devices [5, 21], and unavoidable, it may be argued that in typical situations latency has a bigger impact on performance. However, in the situations where both latency and dropouts are present, their impacts combine, leading to a more pronounced drop in throughput.

Even more interesting is the dropout percentage measure, which seems to show an (almost) perfectly linear inverse relation between dropout percentage and throughput, see Figure 8. For low dropouts of up to 12% there is a small, yet statistically significant, 10% drop in performance with respect to the no-dropout condition. However, there is no statistical difference between dropouts of 6 and 12%. This leads to the question where and how this transition actually happens. Unfortunately, we did not have enough data points in this region to answer this in the first experiment.

Another interesting observation can be made from Figure 7. Data points corresponding to “matched” pairs of dropout duration and intervals e.g., (125, 500), (250, 1000), and (500, 2000), correspond all to the same dropout percentage. However, their throughputs are not equal. This indicates that dropping short bursts frequently is not as detrimental to performance as missing longer chunks infrequently.

Taking this to an extreme leads to a second question. One can envision a situation, in which only every other sample of a 125 Hz sampling USB mouse is taken. Such a device will output only 50% of the samples; however, one would expect the input device will behave like a hypothetical 62.5 Hz mouse, and its performance will be a rather mouse-like and not degraded as much as Figure 7 might indicate.

From Figure 8 it can also be seen that dropouts affect throughput proportionately more at lower latencies. At high dropouts, latency plays a lesser role. This may indicate that the effects of latency and dropouts are both (approximately multiplicative) factors for throughput.

The intent to answer the two questions posed above led us to develop a follow-up experiment.

EXPERIMENT 2
Like the first experiment, this experiment is also based on the ISO 9241-9 standard. But here, we investigate the effect of lower dropout percentages more thoroughly, to determine whether infrequent dropouts still have a measurable effect on throughput. Also, we aim to determine
if there is a threshold for dropout duration, after which the throughput starts to drop progressively.

**Participants**
Twelve students from the local university participated in the experiment, with ages of 21 to 32 (mean 26). Ten were male. All were right handed, or otherwise used the mouse with their right hand. The study lasted 25 to 35 minutes.

**Apparatus and Procedure**
The apparatus, the software, and the procedure in this experiment were essentially identical to the ones of the first experiment. Minor modifications were made to the software to incorporate a different set of conditions, as well as to log some additional service information, such as the actual times of the (now more infrequent) dropout events.

**Design**
The experiment was *within subjects*, and the order in which the combinations of the factors were presented was randomized (without replacement), to compensate for asymmetric transfer of learning effects.

This experiment had four independent variables in a $4 \times (5 \times 5 + 1) = 4 \times 26$ arrangement, for a total of 104 combinations:
- Latency (constant): 10*, 40, 100, and 160 ms;
- Dropout duration: 0*, 10, 20, 40, 80, 160 ms;
- Dropout percentage: 0*, 1, 2, 5, 10, 20%.

In the above list, * denotes the baseline condition, i.e., minimum latency, no latency jitter, and no dropouts. Dropout percentages were modeled via changing the arrival rate in the Poisson process, based on the current *duration*.

As before, six target indices of difficulty (ID), ranging evenly from 2.44 to 5.76 bps, were randomly matched with each of the 104 combinations above.

The dependent variable was effective device throughput (in bits per second), calculated as described earlier.

Each participant completed a set of 104 rounds with different latencies, dropout durations, and dropout percentages. Given that there were 12 participants and 12 recorded target clicks per round, this gave a total of $104 \times 12 \times 12 = 14,976$ trials.

**Results**
Results were analyzed using repeated measures ANOVA. There were significant main effects for latency, dropout percentage, and duration on effective throughput.

**Latency**
The effect of latency on throughput was significant, $F_{3,33} = 359.40$, $p < .0001$. The interaction between latency and dropout percentage was also significant, $F_{15,165} = 1.75$, $p < .05$. No other significant interactions were observed. Figure 9 illustrates the results.

![Figure 9. Throughput for varying levels of latency. Note the non-linear X-axis.](image)

**Dropout Duration**
The effect of dropout duration on the throughput was significant, $F_{5,55} = 2.86$, $p < .05$. According to a Tukey-Kramer test, only the 160 ms condition was different from the others. Figure 10 illustrates the results.

![Figure 10. Throughput for varying dropout durations and percentages.](image)

**Dropout Percentage**
The effect of dropout percentage on the throughput was significant, $F_{5,55} = 16.55$, $p < .0001$. According to a Tukey-Kramer test no statistically significant difference exists between the 0, 1, 2, and 5% conditions. The interaction between the dropout percentage and duration was also significant, $F_{16,176} = 2.18$, $p < .01$. No other significant interactions were observed. Figure 11 illustrates the results.
**DISCUSSION**

For low latencies, below approximately 40ms, we observe no significant differences in throughput, consistent with the first experiment and a previous study [21]. The significant interaction between latency and dropout percentages seems to be due to the 20% dropout condition, which has a significant drop of performance, \( F_{1,11} = 8.17, p < .05 \), even at low latencies, whereas the lower dropout conditions don’t have such behavior, \( F_{1,11} = 0.09 \), ns; see Figure 9.

**OVERALL DISCUSSION**

All results around latency results match previous work. Latency variations, at least with the normal distribution and parameters we used, do not seem to be a deciding factor in interactive systems design. Moreover, we can hypothesize that a higher, yet constant, latency could result in worse performance compared to just keeping the latency variations at their original level. One surprising consequence of this is that filtering may not be as beneficial as commonly assumed, or even detrimental. However, this is a very complex question due to variety of potential distributions and filter parameters.

For small dropout durations (up to 40 ms), dropout percentages can be relatively large (up to 20%), without noticeable effects on performance. On the other hand, longer dropouts (e.g. 160 ms) have significant effects even at low dropout percentages (5% and larger).

This had direct consequences for interactive system design. Consider, e.g., an interactive table with a 30 Hz camera system. Due to the use of a projector and unavoidable delays in camera technology we expect base latencies of at least 120 ms. This already yields a 30% drop in performance relative to a desktop system. Missing two consecutive frames in video processing corresponds to a dropout percentage of 6.6% and should result in an additional drop of 10% in throughput, see Figure 8.

So far, it has been assumed that when input devices miss a sample, potential effects are short-lived. However, in the presence of filtering this may not be the case, as many signal-processing filters rely on several samples to give a stable output. For example, computing object velocity requires at least two frames, whereas computing acceleration requires at least three. Thus, a single dropout will induce longer drops in derived values.

Further source of dropouts are network related delays in (re-)establishing connections. For example, mobile network interfaces incorporate power saving features, and the initial latency of the network communication immediately after the wake-up is substantially larger than typical latencies thereafter. This long initial delay is perceived as a dropout.

Filtering is thought to improve the performance in the presence of the dropouts. However, extrapolating long dropouts (beyond hundreds of milliseconds) is unrealistic, as such durations are similar to typical movement times in the interactions. Since typical movement times are short, predictions then overlap into the next movement and will potentially disrupt it.

**CONCLUSION AND FUTURE WORK**

We presented two user studies examining the effects of device characteristics on 2D pointing tasks. In particular, we examined the effect of latency, latency jitter, and dropouts. While both latency and dropouts have detrimental effect on pointing performance, normally distributed latency jitter seems to have no noticeable effects. Hence, filtering in order to combat latency jitter may actually be harmful, as the filter-added latency may outweigh any potential advantages.

While long dropouts have a dramatic impact on performance, they are encountered in fewer situations, and, overall, their impact on performance is either similar to, or lighter than the impact of frequently encountered latency levels.

The initial indications are that interpolating dropouts by filtering may be of little or no use: for short intervals – because short dropouts have little effect on performance,
and for large dropouts – due to this not being feasible. Further evaluations can verify that.

It is estimated that both latency and dropout duration are multiplicative factors for predicting the throughput. This suggests incorporating them into a homogeneous model for estimating the human pointing performance in the presence of latency and dropouts. This is a subject of future research.

ACKNOWLEDGEMENTS
[Removed for anonymous version].

REFERENCES
12. Han, J. Y., Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection UIST 2005.
26. Ware C., and Balakrishnan, R., Reaching for objects in VR displays: lag and frame rate. ACM TOCHI 1, 4, 1994, 331-356.