Target Following Performance in the Presence of Latency, Jitter, and Signal Dropouts

Andriy Pavlovych Wolfgang Stuerzlinger
York University, Toronto, ON, Canada
{andriyp|wolfgang}@cse.yorku.ca

Abstract

In this paper we describe how human target following performance changes in the presence of latency, latency variations, and signal dropouts. Many modern games and game systems allow for networked, remote participation. In such networks latency, variations and dropouts are commonly encountered factors.

Our user study reveals that all of the investigated factors decrease tracking performance. The errors increase very quickly for latencies of over 110 ms, for latency jitters above 40 ms, and for dropout rates of more than 10 %. The effects of target velocity on errors are close to linear, and transverse errors are smaller than longitudinal ones. The results can be used to better quantify the effects of different factors on moving objects in interactive scenarios. They also aid the designers in selecting target sizes and velocities, as well as in adjusting smoothing, prediction and compensation algorithms.

Keywords: Latency, latency jitter, dropouts, tracking, moving target.

Index Terms: H.5.2. [Information Interfaces and Presentation]: User Interfaces – Theory and methods; evaluation/ methodology.

1 Introduction

Following or acquiring a moving target is a common activity in human computer interaction, especially in entertainment applications and has been studied for quite some time [9, 11, 18, 21, 23]. Many games include animated objects, with which the user sometimes can interact using a pointing device. Following or clicking on a moving object can change the behaviour of the object (e.g., a player in a soccer game), or may be counted as an achievement in itself (military scenarios where "hits" are important).

Many modern games are becoming increasingly collaborative, with collaboration typically taking place over data communication networks. While this network component is routinely affected by multiple factors, there is little evidence that user interaction design takes this system variability into account. Among the factors present in the networked systems are communication latency, the instability of latency, and packet losses. These factors range from cooperative in local wired networks, to potentially quite hostile in wireless cellular segments, and may exhibit large variations over time, e.g., in the Internet. In this paper we investigate the effects of latency, latency jitter (time-domain jitter), dropouts and target speed on tracking accuracy.

1.1 Latency in Computing Systems

Latency can be defined as the time from when the device is physically moved, to the time the corresponding update appears on the screen, or as the delay in device position updates [10]. It has been established that latency and spatial-domain jitter adversely affect human performance in both 2D pointing tasks with stationary targets, in experiments resembling the ISO 9241–9 tapping task [12, 15, 20, 25], as well as in 3D pointing [7, 16, 24, 27]. Latency, for example, manifests itself in larger movement times and correspondingly lower device throughputs, while spatial jitter decreases accuracy. Latency and update rate have both been shown to impact task performance or user response, e.g. [2, 7, 8, 16]. In one of the studies, investigating the effect of network latency in a multiplayer game [19], 50 ms delay was virtually imperceptible, 100 ms was noticeable, and reduced the realism slightly, but was not objectionable, 200 ms was clearly visible, making the play unrealistic, but still possible, and 500 ms delay was absolutely unusable. MacKenzie, Ware [15] looked at latency in pointing and found that 225 ms latency yields very high error rates. Our experience also indicates that latencies greater than 200 ms make interaction uncomfortable and unusable for competitive play. In some cases, latency can also cause nausea and simulator sickness in virtual reality setups [4, 6].

Latency is present to some degree in virtually all systems. As an example, detecting a mouse motion with an optical sensor takes a few milliseconds, sending this information to a host computer takes a few more milliseconds or more if network communication is involved; the operating system may need time to "wake up" the application, which has been waiting for input, the processing of the data itself will then require some time, and, finally, displaying the result will require again a bit of time. In a modern desktop computer with an optical mouse all these delays add up to a total end-to-end delay of 30–35 ms [25]. Latencies present in other modern systems range from 67 to 200 ms in gaming consoles [5], and from 20 to 400 ms in wide-area networks, with about 150 ms being the average value [3, 13]. Current round-trip delays across the Atlantic are 100–150 ms between households, with typical jitter of not more than 50 ms RMS[1].

The latency and latency jitter values chosen for the experiment described in the following sections are representative of the values common for both local and networked systems. In local environments the values are dominated by input and output device lag and by processing jitter. For systems distributed over the Internet, they are dominated by network properties.

1.2 Time Jitter

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. The most common causes of time jitter in human-computer interfaces are algorithm related (e.g., when the processing time depends on the current number of objects present in the game scene), network-related, or operating system scheduling-rated. The effects of *spatial* jitter on 2D pointing have been investigated too, [20].

[1] The network latencies quoted here are based on the statistics collected during 2009–2011 with the help of [13] and other common network utilities.

1.3 Dropouts

Signal losses, like noise, affect many systems. Such losses can happen for various reasons such as: packet losses in the Internet (e.g., the UDP protocol often used for network gaming does not attempt to reconstruct lost packets, mainly for performance reasons, and even TCP packets may be dropped) acoustic trackers might be affected by transient sounds in the environment, electromagnetic trackers may stop working near ferromagnetic objects, and an optical system can be affected by obstructions, poor lighting, image sensor noise. If the signal loss is temporary, it causes a temporary signal *dropout*. During this time, pointing and tracking is impossible, as e.g., the cursor will freeze for a moment. While there are ways to handle such situations, the negative effects of dropouts cannot be fully eliminated. The length of dropouts varies, depending on their nature. For network dropouts it is usually well under a second. Very large variations of latency can also manifest themselves similar to dropouts. The only distinction between very large occasional delays and dropouts is that in the former case the data eventually arrives (although it is often useless by that time) and in the other case is lost completely.

1.4 Lissajous Curves

We used Lissajous curves for modeling the trajectories of our followed targets. Lissajous curves are graphs of a system of the two parametric equations:

$$x = A \cdot \sin(a \cdot t + \varphi)$$
$$y = B \cdot \sin(b \cdot t)$$

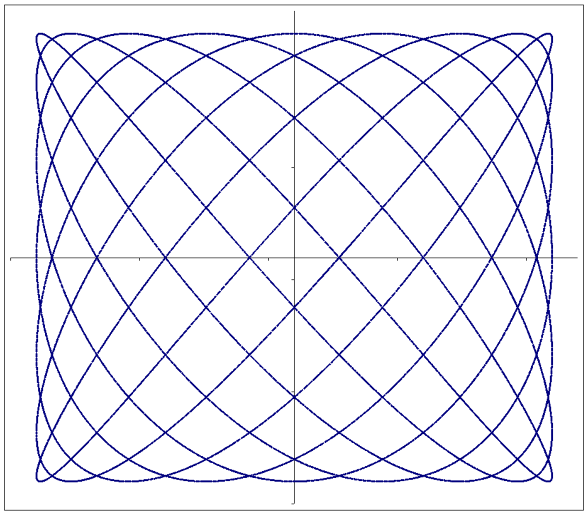

Figure 1. One of the Lissajous Curves used in the experiments. Participants followed *a part of* this curve during the study.

The ratio of the curve frequencies *a* and *b* affects the appearance of the curves. For equal frequencies and amplitudes, and a phase of π/2, the curve becomes a circle. For other ratios, the curve becomes a complex harmonic pattern. One of the patterns used in our experiment is shown in Figure 1. While such curves are easy to model mathematically, the motion of an object following the *invisible* curve is less predictable than a simple one-dimensional oscillation. And, unlike completely random motions, the spatial frequency distribution of Lissajous curves is well-defined and limited, making it easy to separate desired and undesired components in the human response via a frequency domain filter.

At the same time, Lissajous curves mimic a large class of tracking actions. These curves have both slower sections with high curvature and faster ones with low curvature. Also, the spatial frequency of the curves varies, and this makes the curves reasonably similar to real object movements. Real objects obey the laws of physics (such as inertia, where object mass limits acceleration) and conform to the world (real road bends have varying curvature, i.e. varying spatial frequencies). In other words, moving at a constant speed along varying curvatures is difficult and unnatural. Hence, even computer games have to animate objects realistically. Otherwise, they quickly become unplayable – due to the effect of human experience with the real world!

Another argument for Lissajous curves is that for weapon aiming, the user has to keep the target under the cursor or for airplane tracking in the center of the screen. Assuming continuous compensation, Lissajous curves are a good approximation to both motions. Finally, such curves artificially constrain the range of action to a square (or rectangular) region. This corresponds to what occurs in many actual scenarios, as the area available for tracking is naturally limited by the boundaries of a display.

1.5 Task Considerations

One of the considerations was whether to use a tracking task or an interception task. Interception involves the extra action of a button press. We chose to avoid this, as it would add a source of variability, the timing of the click. Thus, we investigated mouse motion separately, consistent with other tracking studies (e.g., [23]). Besides, numerous game interactions do not require mouse clicks. E.g. a “one-shot” gun uses targeting/interception, but a machine/ray gun requires tracking. Also, most rocket launchers require tracking to activate “lock-on”. In sports games, passing to another player also involves a form of tracking. In flight or driving simulators following another entity is also a tracking task.

Tracking tasks can be further subdivided into *pursuit* tracking, in which a target moves and has to be followed with a cursor; and *compensatory* tracking, which involves an evasive target, which the user has to keep it in the center of the screen. Both tracking tasks are similar, and pursuit is considered somewhat easier [23].

1.6 Steering Law

A steering task is the task of moving through a “tunnel” – a trajectory that is constrained on both sides. According to the steering law [1], the time to complete such a task T_c, increases linearly whenever the length *s* increases or the width *W* decreases. Intuitively, one might think that this law applies to a target following task. However, this is not the case. Firstly, the steering law assumes an ideal task, a task in which the boundary is never crossed. Secondly, it does not analyze the performance in the longitudinal direction. That is, if one is always ahead of the tracked target and not crossing the tunnel boundaries, the task appears to be the same as the ideal one. Finally, the steering law may not hold for very wide tunnels, as in such cases only a small fraction of the width will typically be used [14].

Despite these issues unique to steering, it seems logical that increasing speed, the width of the target in a tracking task (or a deviation of the tracking cursor from the tracked target centre) should still increase roughly proportionally, similar to what the steering law predicts. Indeed, this is one of the effects that we will observe in our experiments.

2 Characterizing System Latency and Time Jitter

As there has not been any work on a systematic characterization of tracking performance in the presence of different levels of latency, time jitter and dropouts, we decide to investigate this in a user study. The setup was a modified version of our earlier setup used for measuring performance of acquiring static targets [20]. Before commencing the experiments, we confirmed the properties of our experimental platform, such as the latency and the latency jitter.

We never observed dropouts in our system, thus the only dropouts possible were the ones artificially added during the experiment.

2.1 System Latency and Time Jitter

We used a variation of Mine's method [17] to characterize the end-to-end latency of the mouse used as input device, the computer, the simulation system, and the CRT monitor. We used a CRT, as this technology has notably lower latencies than LCD monitors. The baseline latency of our system was measured at 20±4 ms. We performed three more measurements to verify that the system latency increased by corresponding amounts when an additional, artificial, lag of 30, 90, and 150 ms was introduced via experimental software. The measurements confirmed that.

The base latency value is about 10 ms higher than reported in [20], and we attribute most of this to the use of double buffering for graphics display, which is critical for outputting animated objects. Nevertheless, the observed latency can be considered very low, compared to the latencies present in many modern systems, and especially compared to the latencies present in non-local networks [13]. We also looked at the latency jitter behaviour of our system and found it to be negligible, as in [20].

3 Experiment

The purpose of our study is to investigate how the tracking accuracy degrades, when there is latency, latency jitter, and dropouts present in the system. Once we know the baseline parameters of the system, we can introduce different amounts of these independent factors and observe how the user accuracy changes for targets moving at different speeds. We chose to use Lissajous curves to simulate the movements of targets in part due to their simplicity and ease of analysis, as mentioned above in the corresponding section. However, these curves can follow reasonably well, and (at least) in parts of the motions correspond also to tracking tasks encountered in games, as they include a mix of mostly straight segments with different corner curves.

3.1 Participants

Twelve students from the local university participated in the experiment, with ages ranging from 19 to 32 (mean 26). Nine were male. All were right handed, or otherwise used the mouse with their right hand. The study lasted about 50 minutes.

3.2 Apparatus

The computer was an Intel Pentium-based desktop with a 21″ CRT display, running at a resolution of 1024 × 768 pixels with a horizontal refresh rate of 120 Hz. The mouse used was a 400 dpi Microsoft *Wheel Mouse Optical*, connected via USB and polled at 125 Hz. The software, written in C#, and run under Microsoft *Windows XP SP3* OS, implemented a task of a moving target that had to be followed with a mouse. The width of the target was 35 pixels (0.55″, 14 mm, or ~1.3 degrees of visual field at 60 cm viewing distance). The target trajectory was a Lissajous curve with a width of 640 pixels (10″, 25.4 cm) in each dimension, with four different average frequencies. The individual horizontal and vertical frequencies were chosen in such a way so that the repetition period was substantially longer than the duration of the trial, while each of them was close to the average frequency quoted in the results. The motion of the cursor was delayed (buffered) for the time associated with the current condition. In situations where latency jitter was present it was distributed normally. Dropouts were generated by dropping an appropriate number of buffered samples, treating such events as a Poisson process with an appropriate average arrival time. The software logged the target and the mouse trails, the parameters for all of the independent variables, and also computed various error metrics on the fly. Storing the mouse and target trails enables post-experiment analysis not foreseen before the experiment. During the experiment the users were able to pause their sessions via a space key, which let the current (7-second-long) trial continue until the end and then suspended the program. After a short break the users could resume the study with the same space key.

3.3 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 follow the moving targets as accurately as possible. The participants were informed about the pause functionality of the program and were encouraged to use it any time they felt the need for a short rest, as the task was highly repetitive and of low entertaining value. None of the participants reported any noticeable fatigue and the end of the test. Figure 2 shows two screenshots of the running program.

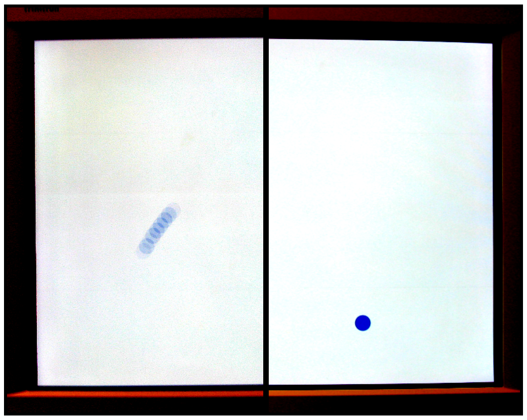

Figure 2. Photograph of the running program output. Left: Target in motion at 32 cpm, blur caused by 1/8 s exposure time of the camera. Right: stationary target during a pause.

3.4 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 possible asymmetric transfer of learning effects. As mentioned earlier, we used a *pursuit* tracking task, where the target moved and had to be followed with the cursor.

3.4.1 Independent Variables

This experiment had five independent variables in a (1 + 2 + 3 + 4) × (3×3 + 1) × 4 = 10 × 10 × 4 arrangement, for a total of 400 combinations:

- *Latency* (constant): 20*, 50, 110, and 170 ms;
- *Latency jitter* (normally distributed, in addition to the constant value above):
    - σ = 0* ms for 20 ms latency,
    - σ = 0, ±20 ms for 50 ms latency,
    - σ = 0, ±20, ±40 ms for 110 ms latency,
    - σ = 0, ±20, ±40, ±60 ms for 170 ms latency;
- *Dropout duration*: 0*, 40, 80, 160 ms;
- *Dropout percentage*: 0*, 5, 10, 20%;

- *Target speed*[2]: 8[3], 16, 24, 32 cycles per minute.

In the above list, * denotes the baseline condition, i.e., minimum latency, no latency jitter, and no dropouts.

Dropouts were intended to complement the jitter behaviour of the systems for cases where jitter is extremely high. Again, the values chosen for modeling dropouts are based on what is considered useable: networks, where 20 % of the packets are not reaching the destination, are borderline useless for real-time interactivity. Dropout percentages were modeled via changing the arrival rate in the Poisson process, based on the current *duration*.

Finally, target speed was the average of the two frequencies used to generate the Lissajous path. More precisely, the two frequencies were ±1 Hz relative to the average frequency.

3.4.2 Dependent Variables

Measuring positional accuracy of following a moving target was one of our primary goals. We chose to divide the error distance (i.e., the instant Euclidean distance between the centre of the target and the tip of the mouse cursor) into two components: a transverse one and a longitudinal one (Figure 3).

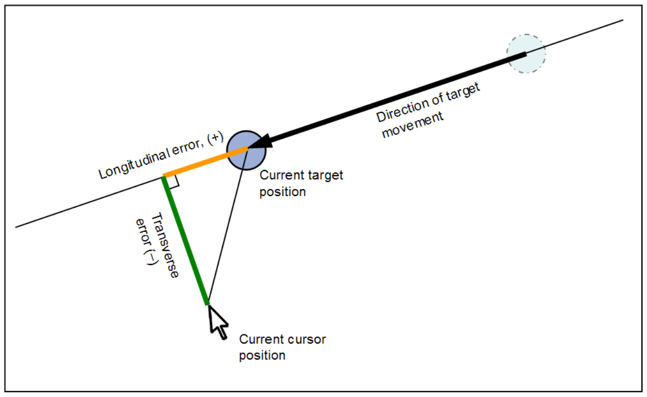

Figure 3. Diagram illustrating the two types of errors measured in the experiment.

The transverse component indicates how far off the *target path* the cursor deviates, and the longitudinal – how far *ahead* or *behind* the cursor is. This separation was done for the following two reasons: the targets are often non-symmetric, but are aligned with the movement direction (e.g., a bus on the road), highlighting the value of computing errors in at least two dimensions; the probability distribution function is normal for both orthogonal components (as was verified subsequently), but it would not be normal for the Euclidean distance, invalidating some of the important assumptions needed for a statistical analysis. After computing the errors for each target position, we compute their RMS values over each of the trials. These are the values that are reported in this paper. The first second of every trial was discarded during the calculations, as it corresponded to the transition from one set of the experimental parameters to the other set. This necessitated a reacquisition of the target that possibly started to move along a different trajectory, a process which typically lasted a few hundred milliseconds.

[2] Although we refer to this variable as *speed*, technically, it is *frequency*, measured in cycles per time interval. We will only use term *frequency* in plots and *speed* elsewhere in the text.

[3] 8 cycles per minute corresponds to about 10 target widths per second *peak* target velocity, or about 6.5 widths *on average*, based on the sinusoidal motion pattern, and the width and amplitude values used in the experiment.

Another dependent variable was *response delay*, indicating the delay between the target movement and the user response (see Figure 4). Even though it is related to the tracking error, there is a notable difference: the response delay mainly looks at *direction* changes in the response. In other words, constant offsets and possible differences in amplitude of the movements have limited effect on this metric, although they affect the tracking errors. Low tracking errors imply low response delay, but not the other way around. As with the tracking errors, we excluded the first second of every trial from the analysis.

Each participant completed a set of 400 rounds, 7 seconds each, with different latencies, dropout durations, dropout percentages and target speeds. The tasks were only replicated across users, to make the duration of the experiment manageable. Given that there were 12 participants, this gave a total of 400×12 = 4800 trials. The 7-second duration of the trial was a compromise determined via pilots, It reduces the variability to acceptable levels, while keeping the total duration of the experiment reasonably short.

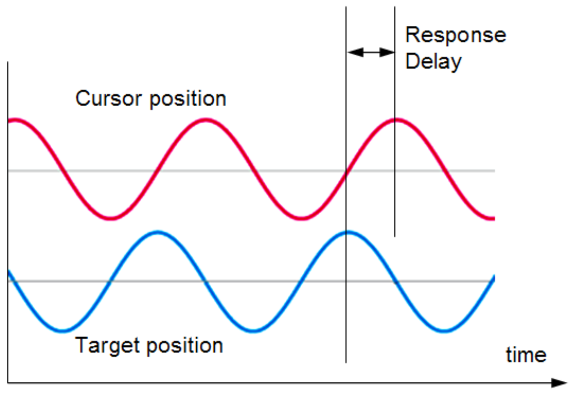

Figure 4. Diagram illustrating response delay as measured in the experiment (note: actual trajectories are not as smooth).

4 Results

The RMS error and the response delay were computed from the target and mouse motion trail logs. For the response delay, we computed the cross-correlation of the mouse trails with the target trajectories in the time domain, to find the time offsets between them, i.e., how far ahead or behind users were relative to the moving target. The results were then analyzed using repeated measures ANOVA. There were significant main effects on tracking error and on response delay for all of the independent variables. Also, there were some interactions between them.

4.1 Main Effects on RMS Tracking Error

4.1.1 Latency

The main effect of latency on longitudinal tracking error was significant, $F_{3,33} = 85.44$, $p < .0001$. According to a Tukey-Kramer test, all pairs were confirmed to be significantly different, except the 20-50 ms and 20-110 ms pairs. The interaction between latency and target speed was also significant, $F_{9,99} = 42.61$, $p < .0001$. So was the interaction between latency and dropout percentage, $F_{9,99} = 2.49$, $p < .05$.

The situation is different for transverse errors. The main effect of latency on transverse tracking error was significant, $F_{3,33} = 67.06$, $p < .0001$. According to a Tukey-Kramer test, only the 170 ms condition was significantly different from the rest; the means of the other three conditions were about 3 pixels apart. The interaction between latency and target speed was significant, $F_{9,99} = 3.28$, $p < .01$, although the effect was not as evident as with the longitudinal error. No other significant interactions were observed for transverse errors. Figures 5 to 7 illustrate the results.

4.1.2 Latency Jitter

The main effects of latency jitter on longitudinal tracking errors were significant at latencies of 110 ms and above, $F_{2,22} = 24.84$, $p < .0001$, $F_{3,33} = 27.50$, $p < .0001$. According to a Tukey-Kramer test, all pairs are statistically distinct, except for the 0-20 and 20-40 pairs of latency jitter in the 170 ms base latency case. Figure 8 shows the case of the 170 ms latency as an illustrative example. The interaction between latency jitter and speed was significant, but only for the 170 ms condition, $F_{9,99} = 2.89$, $p < .01$.

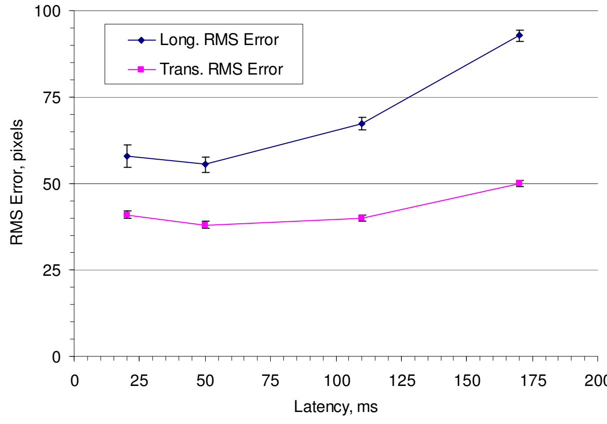

Figure 5. Tracking error for varying levels of latency. Here and below, error bars represent standard error.

For transverse errors, the main effects of latency jitter ware significant for the same base latencies ($F_{2,22} = 8.63$, $p < .01$, $F_{3,33} = 10.02$, $p < .01$). The magnitudes of the differences were noticeably smaller. No other significant interactions were observed. Figures 8 and 9 illustrate the results.

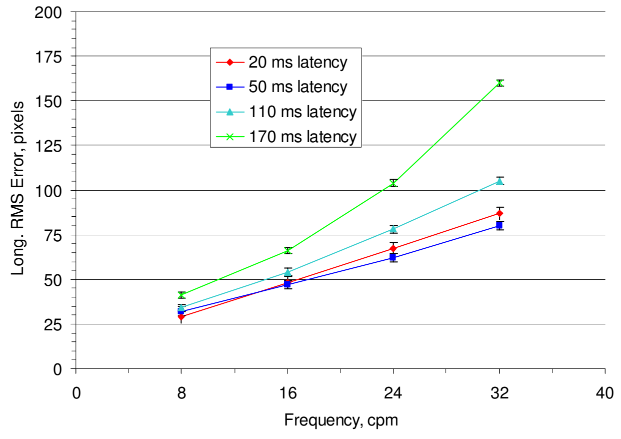

Figure 6. Interaction between latency and target speed for RMS Errors.

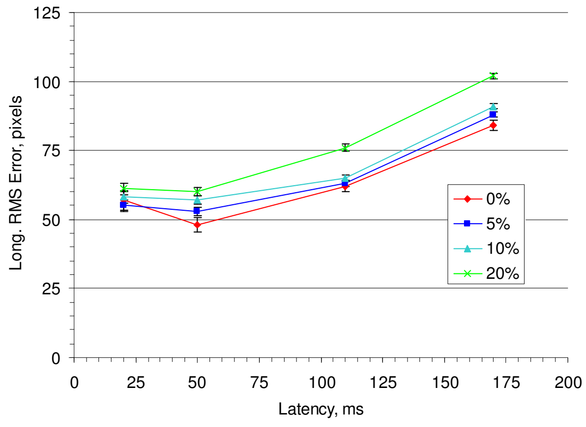

Figure 7. Interaction between latency and dropout percentage for RMS Errors.

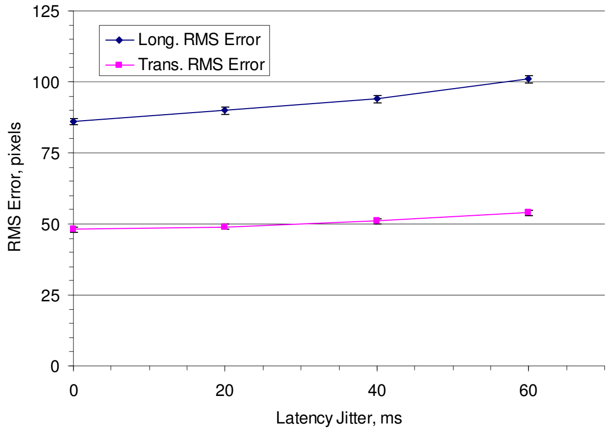

Figure 8. Tracking error for varying levels of latency jitter at 170 ms base latency.

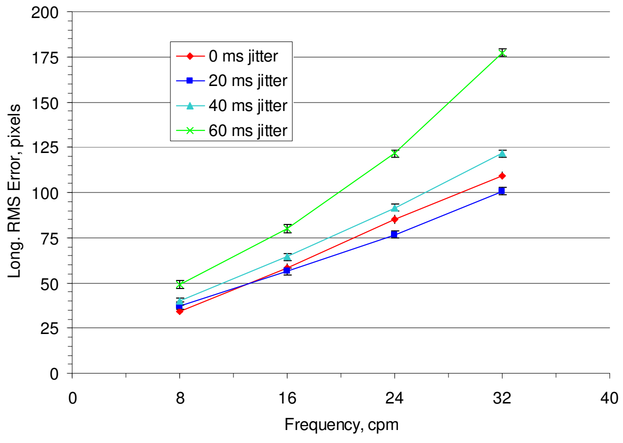

Figure 9. Interaction between latency jitter and target speed for RMS Errors at base latency of 170 ms.

4.1.3 Dropout Duration

The effect of dropout duration on longitudinal tracking error was significant, $F_{3,33} = 53.41$, $p < .0001$. According to a Tukey-Kramer test, there were significant differences between all pairs of conditions, except for the 0-40 pair. The errors increased linearly from 66 to 81 pixels. The interaction between the dropout duration and target speed was also significant, $F_{9,99} = 4.22$, $p < .0001$. Figure 10 illustrates the results.

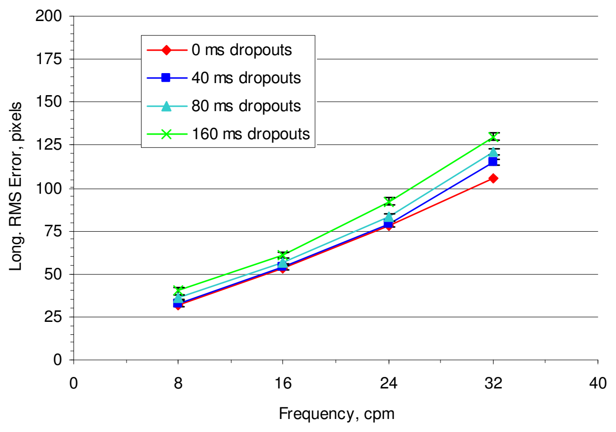

Figure 10. Interaction between dropout durations and target speed for RMS Errors.

For transverse errors, the main effect of dropout duration was significant, $F_{3,33} = 15.98$, $p < .0001$. The errors increased linearly from 41 to 45 pixels. The interaction between the dropout duration and speed was also significant, $F_{9,99} = 2.09$, $p < .05$, with behaviour resembling that of longitudinal tracking errors on Figure 10.

4.1.4 Dropout Percentage

The effect of dropout percentage on longitudinal tracking error was significant, $F_{3,33} = 118.70$, $p < .0001$ (Figure 12). According to a Tukey-Kramer test, there was a significant difference between all pairs of conditions. The errors increased almost linearly from 68 to 82 pixels. The interaction between the dropout percentage and target speed was also significant, $F_{9,99} = 8.06$, $p < .0001$ and similar to the case with dropout duration × speed (Figure 10).
For transverse errors, the main effect of dropout percentage on longitudinal tracking error was significant, $F_{3,33} = 20.79$, $p < .0001$. According to a Tukey-Kramer test, there was a significant only between the 20 % condition and the rest. The errors increased almost linearly from 40 to 48 pixels.

4.1.5 Target Speed

The effect of target speed on tracking errors was significant for both longitudinal and transverse errors, $F_{3,33} = 139.15$, $p < .0001$; $F_{3,33} = 134.63$, $p < .0001$. According to a Tukey-Kramer test, there were significant differences between all pairs of conditions. No new significant interactions were observed. Figure 11 shows the means of the errors.

4.2 Main Effects on Response Delay

Response delay was calculated from the cursor and target trails by computing cross-correlations between them. The cross-correlation coefficient is at its maximum when the waveforms are shifted relative to one another by an amount corresponding to the waveforms' relative time delays.

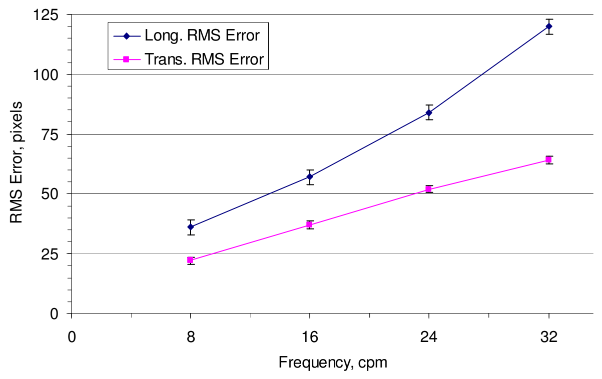

Figure 11. Main effects of target speed on tracking error.

4.2.1 Latency

The effect of latency on response delay was significant, $F_{3,33} = 15.13$, $p < .0001$. According to a Tukey-Kramer test, only the 170 ms latency response was statistically different from the rest. However, if data from one user, whose response lagged the most behind the target, were removed, all pairs were statistically different, except the 20-50 ms and 50-110 ms pairs[4]. The interaction between latency and target speed was significant, $F_{9,99} = 5.75$, $p < .01$. No other statistically significant interactions were observed. Figures 12 and 13 illustrate the results.

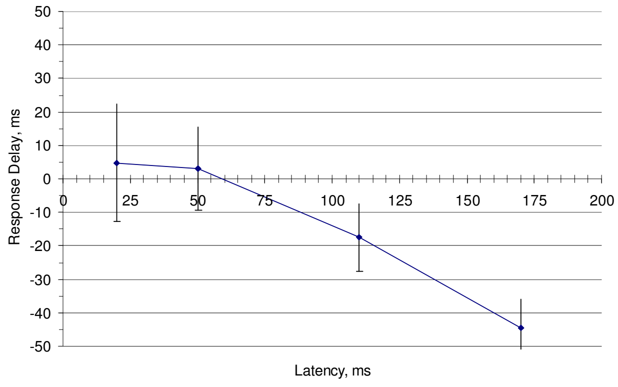

Figure 12. Main effects of latency on response delay.

4.2.2 Latency Jitter

No evidence of the main effect of latency jitter on response delay was observed for any of the base latencies, where the jitter was present (50, 110, 170 ms), $F_{1,33} = 0.48$, *ns*; $F_{2,22} = 1.84$, $p > .05$; $F_{3,33} = 0.78$, *ns*.

4.2.3 Dropout Percentage and Duration

No evidence of the main effect of dropout percentage on response delay was observed, $F_{3,33} = 1.33$, $p > .05$. Also, there was no evidence of the effect of dropout duration on response delay observed, $F_{3,33} = 2.30$, $p > .05$.

[4] There was no sufficient reason to treat that user as an outlier, thus the data was included in all of the other analyses, showing no apparent anomalies anywhere else.

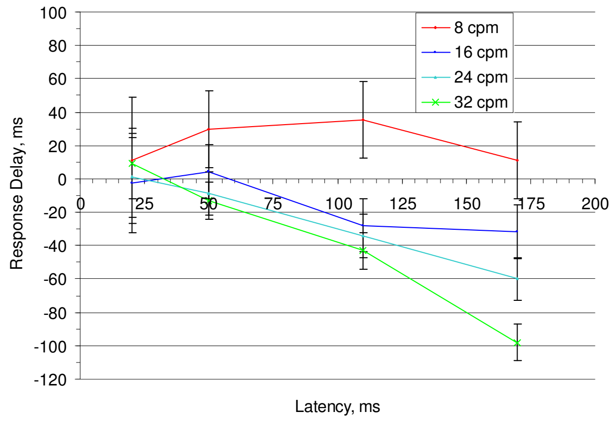

Figure 13. Interaction between latency and target speed for response delay.

4.2.4 Target Speed

The effect of target speed on response delay was significant, $F_{3,33} = 10.48$, $p < .0001$. According to a Tukey-Kramer test, all pairs were statistically different, except the 16-24 and 24-32 pairs of conditions. No new significant interactions were observed. Figure 14 illustrates the results.

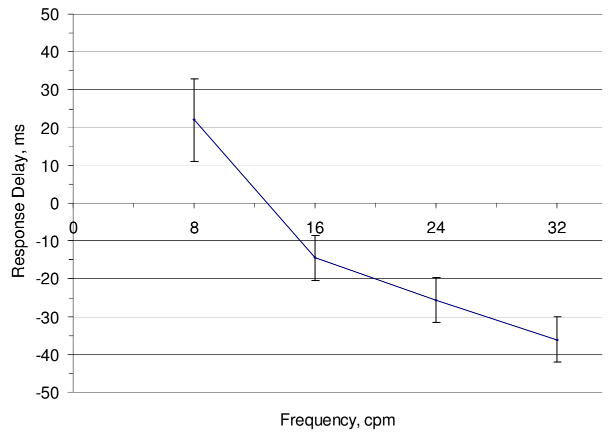

Figure 14. Main effect of target speed on response delay.

5 DISCUSSION

5.1 Maximum Tolerable Latency and Latency Jitter

For low and moderate latencies, up to approximately 110 ms, we observe no significant differences in tracking errors. The significant interaction between latency and dropout percentages seems to be due to the 20 % dropout condition (at which the errors make an abrupt jump). While all of the tested dropout conditions have similar error rates at the lowest latency, at higher latencies the highest dropouts progressively degrade the accuracy, while the lower ones (up to 10 %) do not. From Figure 6 one can see that the errors grow very rapidly with speed at the highest latency of 170 ms, while staying relatively low for lower latencies. Similarly, latency jitter of up to 40 ms can be well tolerated (see Figure 9), while the highest jitter affecting the errors greatly at high target speeds.

A somewhat peculiar observation is the slightly higher error magnitude for the lowest latency, compared with the next latency in sequence. While this difference was not found to be statistically significant, it can be explained by another observation that the human responses follow the target movements very closely at 50 ms latency, lag behind by up to 100 ms for higher latencies, but are slightly *ahead* of the targets at the lowest latency. Apparently, it is this “over-prediction” that causes the error rate to go up slightly at the lowest latency. Even though this difference is not statistically significant, it suggests that, at least in the absence of other factors, a system end-to-end latency of 50 ms is low enough in order not to degrade the performance. This is consistent with the findings of Pavlovych et al. [20], in which the authors saw no difference in mouse performance when pointing at stationary targets in an ISO 9241–9 task. As suggested by one of the reviewers, this effect could be due to 50 ms being the closest to the typical desktop latencies, and it is the latency with which the participants could be the most familiar.

For dropout durations and dropout percentages, the effect on tracking errors seems to be approximately linear. Note that dropouts of 20 % leave only 80 % of the mouse events!

The effect of target speed (or frequency) on the tracking errors is very close to linear. This result is consistent with the other observation in this study that the human responses lag behind the target movements proportionally to the speed and is similar to the effects observed in the past [22]. Effectively, there is a speed-size trade-off in a tracking task. As a consequence, the speed of targets can be controlled during system design, so that the acquisition or miss rate is as desired.

The findings in terms of response delay are mostly in line with the corresponding findings in tracking errors. Among the observations worth highlighting is that the participants can easily accommodate a latency of (up to) about 50 ms, with a response delay of close to zero. After that the response progressively lags behind, reaching 45 ms at 170 ms latency. And while not necessarily directly applicable to design decisions, the response delay behaviour provides an insight into the causes of decreased performance under the influence of latency, in particular.

5.2 Longitudinal vs. Transverse Errors

As we analyzed the transverse and longitudinal errors individually for all of the factors, it will be interesting to note that in almost all of the situations the transverse errors are only 50 to 65 % of the longitudinal (refer to bottom lines on Figures 5, 8, 10, 12, and 14). While the motion of the targets was predictable in the experiment, it appears that sideways accuracy of the users was better. The users could be ahead or stay behind the target, but at least they were close to the target path. Although at this point we can offer no clear explanation for this phenomenon, the implication for interface design is clear: if the targets move, for a given area, the size of the target in the direction of the motion needs to be correspondingly larger. This effect is present even at the lowest tested target speed. However, extrapolating the graph to zero frequency (stationary targets), the difference between the two kinds of errors should disappear, as expected.

5.3 Compensating for Latency and Jitter

There exist solutions to compensate for latency and other factors. For example one could average samples to smooth the target motion or extrapolate movements to compensate for propagation delays of the true signal. While some solutions are viable (e.g. artificially increasing latency to hide latency jitter [26]), others are hard or impractical, like the task of predicting irregular trajectories. As an example, a typical movement time in interactions is hundreds of milliseconds, and this is close to the levels of latencies often affecting the performance. If the parameters of the user motion change from the ones assumed by the prediction algorithm, the predicted position will remain imprecise until an update with a true motion is received and the parameters are updated accordingly. Such update can require at least the [system delay] time. Until that happens, the deviation from the intended position will be proportional to the product of system delay and the difference in predicted and actual movement velocities.

6 Conclusion and Future Work

We presented an experimental evaluation of latency, time jitter, and dropouts on target following, a task that is common to interactive entertainment systems and computer games. Our user study reveals that all of the investigated factors decrease tracking performance. The errors start to increase very quickly for latencies of over 110 ms, for latency jitter levels above 40 ms, and for dropout rates of more than 10 %. The effects of target velocity on errors are close to linear, and, finally, the transverse errors are always smaller in magnitude, compared to longitudinal ones. The results can be used to better quantify the effects of different factors on moving objects in interactive scenarios, such as video games. They also enable designers to make better-informed choices for selecting target sizes and velocities, as well as for adjusting smoothing, prediction and compensation algorithms to constrain the overall system characteristics to the values above, in order to avoid large losses in performance.

6.1 Modelling

In pointing, Fitts' Law predicts the speed of interaction accurately. It appears to be feasible to create a similar model for tracking. However, and based on initial attempts, it is not trivial to combine the effects of multiple factors into a single expression. Furthermore, we believe that the model should not be just a regression on experimental data (such data is readily available in this paper). Investigation of the underlying principles that lead to losses in performance should be the starting point in developing a new model, just like Fitts' Law's relation to information theory principles has led to its wide acceptance.

6.2 Generalizability to other Pointing Devices

Although we investigated only mouse-based input, the findings are likely to generalize to other devices, as latency or signal dropouts will affect them similarly. However, the quantitative effects are likely to vary for different classes of devices. For example, direct touch interfaces are affected differently by latency, as one can immediately see where one points and adjust the trajectory based on non-delayed direct visual observation. On the other hand, an isometric joystick – a relative pointing device – could be more similar to a mouse than to a direct touch panel. However, these conjectures still need to be examined in the future.

References

[1] Accot, J. and Zhai, S. 1997. Beyond Fitts' law: models for trajectory-based HCI tasks. *ACM CHI '97*. 295–302.

[2] 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.

[3] Armitage, G. and Stewart, L. 2004. Limitations of using real-world, public servers to estimate jitter tolerance of first person shooter games. *ACM ACE '04*, 257–262.

[4] Bolia, R. S., Morley, R. M., Nelson, T. W., Roe, M. M. 2000. Assessing Simulator Sickness in a See-Through HMD: Effects of Time Delay, Time on Task, and Task Complexity. *Proceedings of the 2000 IMAGE Conference*, available Online at: http://www.spatiald.wpafb.af.mil/Publications/IMAGE00.pdf

[5] Console Gaming: The Lag Factor. http://www.eurogamer.net/articles/digitalfoundry-lag-factor-article.

[6] Craig, S. J., Reid, L., and Kruk, R. 2000. The effect of visual system time delay on helicopter control. *Proceedings of the IEA 2000/HFES 2000 Congress*, 3-69–3-72.

[7] 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.

[8] 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, 1999*, 1182–1186.

[9] Fekete, J.-D., Elmqvist, N., and Guiard, Y. 2009. Motion-pointing: target selection using elliptical motions. In *Proc. CHI '09*. 289–298.

[10] Foxlin, E., Motion tracking requirements and technologies. In Handbook of virtual environments: Design, implementation and applications, Lawrence Erlbaum, 2002, 163–210.

[11] Hoffmann, E. R. 1991. Capture of moving targets: a modification of Fitts' Law', *Ergonomics*, 34:2, 211–220.

[12] ISO/DIS 9241–9 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.

[13] just ping – Online web-based ping: remote ping a server with 40 checkpoints worldwide, www.just-ping.com.

[14] Kulikov, S., MacKenzie, I. S., and Stuerzlinger, W. 2005. Measuring the effective parameters of steering motions. *CHI '05 extended abstracts*, 1569–1572.

[15] MacKenzie, I. S., and Ware, C. 1993. Lag as a determinant of human performance in interactive systems. *ACM CHI 1993*, 488–493.

[16] 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.

[17] Mine, M. R., Characterization of End-to-End Delays in Head-Mounted Display Systems. University of North Carolina at Chapel Hill, 1993.

[18] Mould, D. and Gutwin, C. The effects of feedback on targeting with multiple moving targets. *Graphics Interface 2004*, 25–32.

[19] Pantel, L and Wolf, L. C. 2002. On the impact of delay on real-time multiplayer games. In *Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video* (NOSSDAV '02), 23–29.

[20] Pavlovych, A. and Stuerzlinger, W. 2009. The tradeoff between spatial jitter and latency in pointing tasks. In *Proc. EICS '09*, 187–196.

[21] Port, N. L., Kruse, W., Lee, D., and Georgopoulos, A. P. 2001. Motor Cortical Activity during Interception of Moving Targets. *J. Cognitive Neuroscience* 13, 3, 306–318.

[22] Poulton, E. C. 1952. Perceptual anticipation in tracking with two-pointer and one-pointer displays. *British Journal of Psychology*, 43, 222–229.

[23] Poulton, E. C. Tracking Skill and Manual Control. Academic Press (1974).

[24] 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.

[25] 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.

[26] Tumanov, O., Allison, R., and Stuerzlinger, W. 2007. Variability-Aware Latency Amelioration in Distributed Environments, *IEEE VR 2007*, 123–130.

[27] Ware C., and Balakrishnan, R., Reaching for objects in VR displays: lag and frame rate. *ACM TOCHI 1*, 4, 1994, 331–356.