The Effects of Delay on Game Actions: Moving Target Selection with a Mouse

Mark Claypool
Worcester Polytechnic Institute – Computer Science
Worcester, MA, 01609, USA
claypool@cs.wpi.edu

Ragnhild Eg, Kjetil Raaen
Westerdals – Oslo School of Art, Communication and Technology
Box 9215 Grønland, 0134 Oslo
{egrag|raakje}@westerdals.no

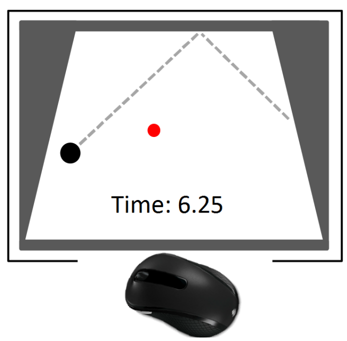


Figure 1: *Puck Hunt* – Player tries to click on moving target (the puck) with mouse (the red ball). Game adds delay to mouse input and varies puck speeds each round.

CHI PLAY'16, Extended Abstracts, October 16–19, 2016, Austin, TX, USA.
ACM 978-1-4503-4458-6/16/10. http://dx.doi.org/10.1145/2968120.2987743

Abstract
In modern computer systems, user input, particularly for computer games, is affected by delay from local systems, networks and servers. While general awareness of the degradation effects of delay on player performance and quality of experience are well known, an understanding quantifying how specific player actions are impacted by delay is missing. This work presents a user study that gathers data on player actions for a range of delay and game conditions for the fundamental game action of selecting a moving target with a mouse. Analysis shows player sensitivity to delays in all conditions, with particular sensitivity when targets are fast. From the data, we derive a simple analytic model that is a promising step towards a broadly applicable tool to better understand and compensate for delay in games and interactive applications.

ACM Classification Keywords
H.5.2 [Information interfaces and presentation (e.g., HCI)]: User Interfaces:Input devices and strategies (e.g., mouse)

Introduction
Real-time games require players to make many time-sensitive actions that can suffer when the computer responses lag behind player input. Even temporal delays as small as milliseconds can hamper the interplay between players' actions and intended results. For example, lag when aiming and firing with a mouse can make it difficult for a player to hit a target in a shooting game, hurting the player's score and degrading the quality of experience.

While there exist many methods to compensate for delays [3], including system-level treatments (e.g., real-time priorities), latency compensation algorithms (e.g., dead reckoning) and even game designs to mitigate delay (e.g., delayed avatar response), an understanding of how latency affects fundamental player actions in games is needed in order to choose the most effective delay compensation techniques.

Moving Target Selection

The task of selecting a moving target with a mouse is common to many PC games. Most notably, the popular first person shooter (FPS) genre (e.g., *Call of Duty*, Activision, 2003) has moving target selection with the mouse as the primary method of aiming and shooting. Likewise, the newer multiplayer online battle arena (MOBA) genre (e.g., *League of Legends*, Riot Games, 2009) uses moving target selection with a mouse for casting spells.

While foundational studies on user actions [5, 8, 9, 11, 12] have shown promise to better understand the effects of modern computer systems on games, such studies have not incorporated delays nor most game interactions (e.g., selecting a moving target with a mouse). Game studies have incorporated delays and do focus on game genres (e.g., first person shooters) [1, 2, 4, 6, 7], but fail to provide an understanding of the effects of delay on isolated player actions in games.

Ideally, game designers and system developers would have a model as far reaching and robust as Fitts' Law [8], an ergonomic model for the time it takes for a user to select a target of a given size at a certain distance, but accurate for fundamental game actions in the presence of delay. Our work takes a first step to providing such a model.

We design and implement a game that isolates a fundamental action, that of selecting a moving target with a mouse (see sidebar), while allowing for control of the delay between the player input and the resulting game action. In *Puck Hunt*,[1] (Figure 1) players try to hit a moving target (the puck) with the mouse, where mouse input is delayed and game difficulty is controlled with target speed. The game is deployed in a user study with over 30 participants, with added delays ranging from 0 to 400 milliseconds and target speeds ranging from 150 to 450 pixels/second.

[1] A pun on the classic game *Duck Hunt* (Nintendo, 1984).

Analysis of the results shows the time to select a moving target with the mouse increases polynomially with delay – this is in contrast to earlier work [10] that showed a linear relationship. The time to select the target does not vary with target speed for low delays, but there are pronounced interaction effects between added delays and target speeds for high delays. User opinions on the quality of experience (responsiveness) show a more pronounced decrease than does performance, even for modest delay increases. Lastly, we derive an accurate analytic model for the average time to select a moving target based on delay and target speed. The model is polynomial with delay and includes a linear interaction term for delay and target speed.

The rest of this paper is organized as follows: the first section describes our methodology to study delay and moving target selection with a mouse, including development of a custom game and a user study; the second section analyzes the results from the user study data and presents our analytic model; and the final section summarizes our conclusions and presents possible future work.

Methodology

To investigate the effects of delay on player actions, we deployed the following methodology: 1) Design and develop a game (*Puck Hunt*) that isolates player actions and controls added delay; 2) Conduct a user study to evaluate the impact of delay on player actions, measuring player performance and quality of experience (QoE); and 3) Analyze the results of the user study and develop an analytic model for player performance and delay.

Puck Hunt

The custom game Puck Hunt allows for study of a single player action – selection of a 2-d, moving target with a mouse – with controlled amounts of delay. The action chosen is one common to more full-fledged games (see sidebar, page 2). To minimize the latency inherent in the

Speed	
Slow	150 pixels/sec
Medium	300 pixels/sec
Fast	450 pixels/sec

Table 1: Puck speeds for user study.

Delay (milliseconds)
0, 25, 50, 75
100, 125, 150, 175
200
300
400

Table 2: Delays for user study.

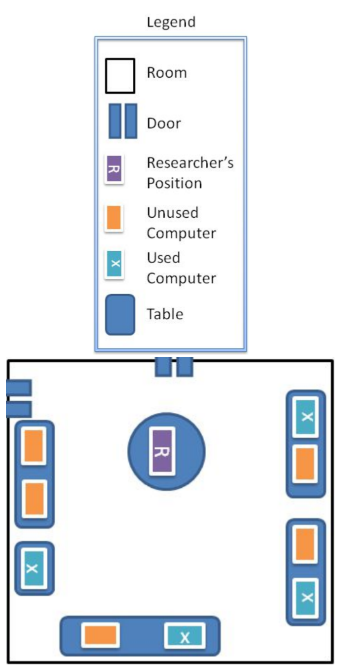


Figure 2: Lab for user study.

software, Puck Hunt is written in C++ using OpenGL with support from the Angel 2D game engine.[2]

In Puck Hunt, the player proceeds through a series of short rounds, where each round has a large black ball, the puck, that bounces around the screen. The player moves the mouse to control the small red ball (i.e., the cursor), and attempts to “hit” the puck by moving the ball over the puck and clicking the mouse button. Once the player has successfully hit the puck, the puck disappears and a notification pops up telling the player to prepare for the next round. Thereupon pressing any key, a new round starts, with the puck at a new starting location with a new orientation and speed. The player is scored via a timer that counts up from zero at the beginning of each round, stopping when the puck is hit.

Each round, the puck moves with one of 3 possible speeds, shown in Table 1. Effectively, these speeds create different levels of difficulty. The game also adds a controlled amount of delay selected from a set of 11 possible values, shown in Table 2. The delay is added to all mouse movements and button clicks for the duration of the round. Each delay & speed combination only appears a fixed number of times – the number of iterations, controlled by a configuration parameter – but the entire set of combinations is shuffled so as to appear in a random order.

Every 30 rounds, the game stops for a minimum of 20 seconds to allow the player to rest/regain concentration, with a countdown timer shown to the player via a popup window.

Exactly once for each combination of delay & speed, the player is asked to rate the quality of experience (QoE) based on the responsiveness during the round, depicted in Figure 3. The game pauses until the player selects a choice, 1–5.

[2] http://angel2d.com/

Puck Hunt runs in fullscreen mode at 1080p resolution (1920x1080 pixels). The puck is 100 pixels in diameter and the mouse cursor (the red ball) is 25 pixels in diameter.

User Study

Our user study was conducted in a windowless computer lab with bright, fluorescent lighting, the layout shown in Figure 2. The computers were Dell PCs with Intel i7-4790 4 GHz processors, 4 GB GeForce GTX 960 graphics cards and 16 GB of RAM, running Microsoft Windows 7. The monitors were 24" Dell U2412M LCDs with a native resolution of 1920x1200 pixels and a refresh rate of 59p Hz.

Participants were solicited through advertising via WPI email lists. Incentives included a raffle for a $25 gift card for participating and a $25 gift card for the user with the highest score. Game development students that participated also received 1 extra point on their final exams.

First, users heard a scripted brief about the study and signed an Institute Review Board (IRB) consent form at the researcher's position (see sidebar Figure 2). Next, users sat at a computer and were asked to make themselves comfortable by adjusting chair height and monitor angle/tilt so as to be looking at the center of the screen. Users were encouraged to shift the mouse to whichever hand was preferred.

Users then filled out a survey coded using Qualtrics,[3] providing their demographics and gaming experiences. After completion of the survey, the game and incentive options were described followed by launching the game.

Play commenced immediately, but the first two rounds were used for practice only and the results were not recorded. Play then proceeded through 5 iterations of all shuffled combinations of delay & puck speed (Table 1 and Table 2), with one QoE question for each delay-speed combination and a forced pause every 30 rounds. In total, users played

[3] https://www.qualtrics.com/

165 recorded rounds each, which took about 15 minutes including answering questions and pausing.

Note, the delays in Table 2 added by Puck Hunt are in addition to any delays inherent in the base computer system. Since such base delays have been shown to be significant [13], we measured the base delay for mouse actions on our lab computers using a Blur-busters type technique[4] (see sidebar). The measurement method was repeated 5 times, resulting in base delay values of 93, 99, 101, 101 and 112 milliseconds. Hence, 100 milliseconds is added to all subsequent delay values.

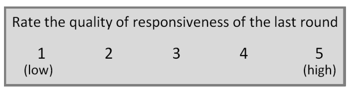
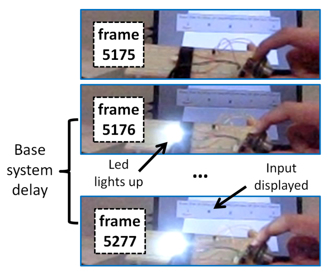


Figure 3: Quality of experience prompt to player.

Measuring Base Delay



Method: A bread board with an led was connected via a wire soldered to a mouse so that the led lit up when the button was clicked. A high frame rate camera (a Casio EX-ZR200) filmed the player clicking on the QoE prompt, recording the action at 1000 f/s. By manually examining the individual video frames, the frame number when the light appeared with the button click is subtracted from the frame number when the QoE prompt shows the input, giving the base delay.

Results

Demographics

Thirty-two users participated in the study. Ages ranged from 18-26 years with a mean and median of 21. Twenty-three identified as male, 8 as female and 1 did not specify. Twenty-seven indicated they were right-handed, 4 left-handed and 1 ambidextrous, but all used the computer mouse right-handed. The mean self-rating as a computer gamer (scale 1–5) was 3.5, showing a slight skew to having "high ability". Self-ratings for PC gamer and network gamer had similar distributions as for computer gamer. Exactly half the users played 6+ hours of computer games per week, about the same fraction that used a computer (PC/Mac) with a mouse 6+ hours per week. Most studied Computer Science, Game Development or Engineering.

Objective – Game Performance

Puck Hunt is designed to isolate the fundamental action of moving target selection with a mouse. As such, we assess Puck Hunt player performance, where the player's score is the time to hit the puck – the lower the number the better.

Figure 4 depicts player performance. The x-axis is the input delay (added delay + base delay) and the y-axis is the time to hit the puck, in milliseconds. There are three trend-lines, one for each puck speed tested. Each point is the mean hit time for all users for that delay & speed combination, shown with a 95% confidence interval. Overall, there is an increase in mean hit time as delay increases. This increase appears polynomial or exponential over the range of delays tested. For delays under 200 milliseconds, the speed of the puck does not impact mean hit time. However, starting at delays of 225 milliseconds (for fast pucks) and 400 milliseconds (for medium speed pucks), the faster speed pucks become harder to hit than the slowest speed pucks. At the extreme (500 milliseconds) delay, the fast pucks take 5x longer to hit than when there is minimal (100 milliseconds) delay and even the slow pucks take over 2.5x longer to hit.

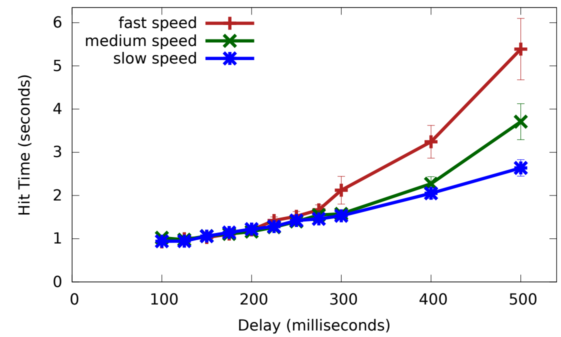


Figure 4: Player performance – Hit time versus delay, grouped by puck speed.

Figure 5 depicts another graph of player performance using the same data but analyzed by speed. The x-axis is the speed in pixels per second and the y-axis is the time to hit the puck, in milliseconds. There are five trend-lines, one for the total delays (added delay + base delay) of 100–500 millisec-

[4] http://www.blurbusters.com/gsync/preview2/

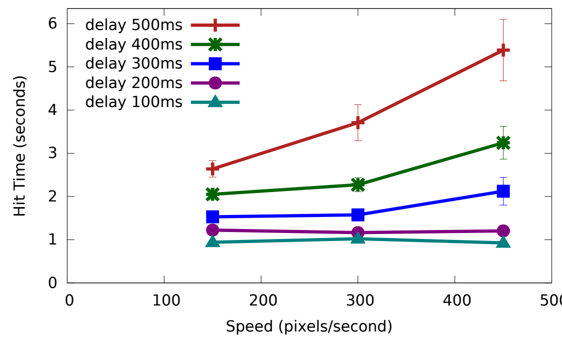


Figure 5: Player performance – Hit time versus speed, grouped by added delay.

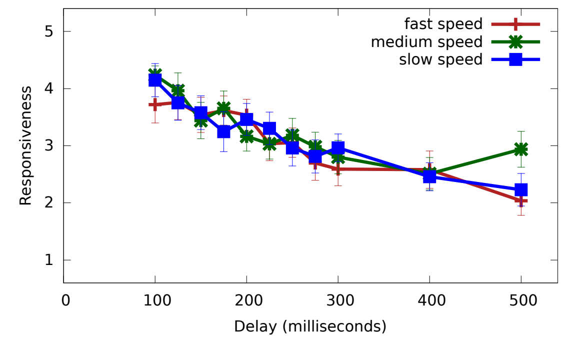


Figure 6: Quality of Experience – Responsiveness versus delay, grouped by puck speed.

onds.[5] Each point is the mean hit time for all users for that delay & speed combination, shown with a 95% confidence interval. Overall, there is an increase in hit time as the puck speed increases. This increase appears mostly linear for the range of puck speeds tested. Delay impacts the hit time for all puck speeds, but is most pronounced for the highest puck speeds as seen by the diverging lines. As seen in alternate form in Figure 5, for delays of 200 milliseconds and under, the lines are flat – the speed of the puck does not impact mean hit time.

Subjective – Quality of Experience

While player opinion often correlates with game performance, subjective measures can ascertain the quality of the experience (QoE) beyond just the score. For Puck Hunt, for each delay & speed combination, users were asked to rate the responsiveness of the game (Figure 3).

Figure 6 depicts a graph of the responsiveness versus delay. The x-axis is the total input delay and the y-axis is the quality of experience – here, the responsiveness of the round. There are three trend-lines, one for each puck speed tested. Each point is the mean rating for all users for that delay & speed combination, shown with a 95% confidence interval. From the graph, there is an observable downward trend in QoE with an increase in delay, indicating players perceive the delays. However, unlike for performance, there is no noticeable separation of QoE with puck speed suggesting players are able to gauge responsiveness based on delay independently of the difficulty of the game action.

Model

While trends in player performance and experience with delay provide valuable insights for researchers and developers, more useful is an analytic model of the relationships. As a step towards such a model, we modeled the mean time to select a moving target with a mouse with delay.

Based on the previous analysis, there is a clear upward trend in mean time with increased delay, possibly linear but more likely polynomial to capture the observed curvature. The time trend with puck speed is less clear – for the

[5]The other delays tested are not shown to keep the graph readable.

range of speeds tested, there is little effect of puck speed on performance with low delays, however there is for high delays. Thus, there seem to be important interactions between speed and delay a model should incorporate.

Thus, we propose modeling the time to select a target with a mouse (T) as a quadratic polynomial for delay only, with an interaction term for delay (D) and speed (S):

$$T = a + bD + cD^2 + dD \cdot S \tag{1}$$

where a, b, c and d are constants determined empirically through user study. Fitting this regression model to our user study data using R yields a fit of R^2 0.95, F-stat 118 and $p < 2.511e^{-16}$, with the simplified final model:

$$T = 1 - 0.005D + 0.00002D^2 + 0.000009D \cdot S \tag{2}$$

T is the mean time to select a target in seconds, D is the total input delay in milliseconds, and S is the target speed in pixels/second.

Note, the final model as presented (Equation 2) likely holds primarily for the conditions tested. The size of the target is known to affect target selection time, most famously studied for Fitts' Law [8]. While target size was not varied in our study, combining Fitts' Law with our results may produce a unified general model. Such modeling should consider both the absolute target size in pixels and also the target size relative to the screen resolution.

Delay and Online Games

The results presented are relevant to all forms of input delay, whether from the local system (e.g., operating system and hardware) or from the network. In particular, the results pertain to cloud games where all player input is sent to the cloud for rendering, meaning all mouse actions, both movement and clicking, are delayed by the local system, network and server. However, traditional network games – where mouse movement is processed and rendered by the local client – have only local delay for mouse movement, but incur additional delays for mouse clicking since the latter has network and server processing delays, too.

Conclusion

Understanding the effects of delay on player input can help game designers and researchers develop solutions to mitigate the negative impact of delay on players. While some previous work has measured the effects of delay on games and other previous work has modeled user input for tasks without delay, there has yet to be thorough exploration and/or models quantifying the effects of delay on fundamental player actions.

This paper presents work in progress towards a model for player actions with delay. We present results of a user study with a custom game wherein players selected moving targets with a mouse with delayed input, the game difficulty controlled by the target speed. Over 30 users participated in the study, providing data for delays from 100 to 500 milliseconds and 3 target speeds – in total, over 5000 observations of player performance. In addition, players provided over 1000 subjective quality assessments for each of the different delay & speed combinations.

Analysis of the results shows a measurable increase in the time to select a moving target even for low delays and a sharp increase in selection time for higher delays and fast targets. While target speed is not a factor for low delays, subjective opinions show users are sensitive to even modest amounts of delay. A derived analytic model provides a good fit for the mean time to select a moving target, with quadratic terms for delay, no terms for target speed, and an important interaction term that captures the effects of target speed combined with delay.

While promising, there are several areas for continued work. Additional models and analysis can be done for mouse clicks and quality of experience. Study of target selection over a wider range of target speeds would help the results pertain to a broader set of games, as would more general models with target size and screen size (distance). Other forms of player input that involves target selection (e.g., analog controller, touch on mobile/tablet) or even keyboard or game controller button pressing could be explored.

Acknowledgments
Thanks to Marco Duran and Matthew Thompson for conducting the user studies and measuring system delays.

REFERENCES

1. Rahul Amin, France Jackson, Juan E. Gilbert, Jim Martin, and Terry Shaw. 2013. Assessing the Impact of Latency and Jitter on the Perceived Quality of Call of Duty Modern Warfare 2. In *Proceedings of the International Conference on Human-Computer Interaction – Users and Contexts of Use*. Las Vegas, NV, USA, 97–106.

2. Grenville Armitage. 2003. An Experimental Estimation of Latency Sensitivity in Multiplayer Quake 3. In *Proceedings of the 11th IEEE International Conference on Networks (ICON)*. Sydney, Australia.

3. Yahn W. Bernier. 2001. Latency Compensating Methods in Client/Server In-game Protocol Design and Optimization. In *Proceedings of the Game Developers Conference (GDC)*. San Francisco, CA, USA.

4. K. Chen, P. Haung, G. Wang, C. Huang, and C. Lee. 2006. On the Sensitivity of Online Game Playing Time to Network QoS. In *Proceedings of IEEE Infocom*. Barcelona, Spain.

5. T. Chiu, C. Lin, K. Young, C. Lin, S. Hsu, B. Yang, and Z. Huang. 2011. A Study Of Fitts' Law on Goal-Directed Aiming Task with Moving Targets. *Journal of Perceptual and Motor Skills* 113, 1 (Aug. 2011), 339–352.

6. Mark Claypool. 2005. The Effect of Latency on User Performance in Real-Time Strategy Games. *Elsevier Computer Networks, Special Issue on Networking Issues in Entertainment Computing* 49, 1 (Sept. 2005), 52–70.

7. Mark Claypool and Kajal Claypool. 2006. Latency and Player Actions in Online Games. *Commun. ACM* 49, 11 (Nov. 2006).

8. Paul M. Fitts. 1954. The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. *Journal of Experimental Psychology* 47, 6 (June 1954), 381–391.

9. Abir Al Hajri, Sidney Fels, Gregor Miller, and Michael Ilich. 2011. Moving Target Selection in 2D Graphical User Interfaces. In *Proceeding of the IFIP TC International Conference on Human-Computer Interaction (INTERACT)*. Lisbon, Portugal.

10. Errol Hoffmann. 1992. Fitts' Law with Transmission Delay. *Ergonomics* 35, 1 (1992), 37–48.

11. I. S. MacKenzie, A. Sellen, and W. Buxton. 1991. A Comparison of Input Devices in Elemental Pointing and Dragging Tasks. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*. New Orleans, LA, USA, 161–166.

12. Michael J. McGuffin and Ravin Balakrishnan. 2005. Fitts' Law and Expanding Targets: Experimental Studies and Designs for User Interfaces. *ACM Transactions on Computer-Human Interaction* 12, 4 (Dec. 2005), 388–422.

13. Kjetil Raaen and Andreas Petlund. 2015. How Much Delay Is There Really in Current Games? *Proceedings of the 6th ACM Multimedia Systems Conference* (2015), 2–5.