WPI-CS-TR-20-05 July 2020

Game Input with Delay - A Model for the Time to Select a Target with a Mouse

by

Shengmei Liu and Mark Claypool

Computer Science
Technical Report
Series

WORCESTER POLYTECHNIC INSTITUTE

Computer Science Department
100 Institute Road, Worcester, Massachusetts 01609-2280

Game Input with Delay - A Model for the Time to Select a Target with a Mouse

Shengmei Liu
sliu7@wpi.edu
Worcester Polytechnic Institute
Worcester, MA

Mark Claypool
claypool@wpi.edu
Worcester Polytechnic Institute
Worcester, MA, MA

ABSTRACT

Computer game player performance can degrade with delays from both the local system and network. Analytic models and simulations have the potential to enable exploration of player performance with delay as an alternative to time-intensive user studies. This paper presents an analytic model for the distributions of elapsed times for players doing a common game action – selecting a moving target with a mouse with delay – derived from results from prior user studies. We develop and validate our model, then demonstrate the use of our model via simulation, exploring player performance in games with different configurations and delays.

KEYWORDS

modeling, distribution, elapsed time, moving target

ACM Reference Format:
Shengmei Liu and Mark Claypool. 2020. Game Input with Delay - A Model for the Time to Select a Target with a Mouse. In *Proceedings of Technical Report (TR'20).* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Computer games require timely responses to player actions in order to provide an immersive interactive experience. Unfortunately, computer input hardware and software always have some delay from when a player inputs a game command until the result is processed and rendered on the screen. Delays on the local computer system are still at least 20 milliseconds and can range much higher, to about 250 milliseconds for some platforms and game systems [11]. Games played over a network, such as multi-player game systems or cloud-based games, have additional delays from the network processing as game data has to be transmitted to and from a server. Both local delays and network delays impact the player, degrading both player performance and player Quality of Experience (QoE) as total delay increases [3–5].

Research that studies delay and games typically involves user studies with participants playing a game with controlled amounts of delay. While these studies can be effective for ascertaining player performance for specific games [1, 2], and specific game actions [15], they are time-intensive, requiring months of time and often can only gather data over the limited range of game and system configurations tested. Moreover, some societal conditions (e.g., social distancing) can make organizing and executing traditional user studies impossible.

As an alternative approach, analytic models of player performance and simulations of computer games can provide for a broad exploration of the impact of delay on game conditions without costly user studies. However, such an approach can only be effective if it accurately represents player performance. Data from studies that isolate "atoms" of game actions have the potential to provide the foundation for accurate analytic models of player performance, and resultant simulations that use them can help explain and predict the effects of delay for a wide-range of games and delay conditions. This paper makes just such a contribution – an analytic model, with demonstrated use in simulation.

We use data gathered from two previous users studies that measured user performance for an atomic game action – selecting a moving target with a mouse. Users played a game that had them select targets that moved around the screen with different speeds and with different amounts of added delay. The studies recorded the elapsed time to select the target, coupled with a player-provided self-rating of skill.

We use the data from these studies in multivariate, multiple regression to derive models of the expected elapsed times for selecting a moving target with a mouse, and the *distribution* of the elapsed times. Used together, the models provide an accurate representation of target selection times over a range of delays and target speeds and two levels of player skill. We demonstrate the use of the models in analytic models of player performance and simulations of basic

TR'20, July 2020, WPI, Worcester, MA, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

shooting games to predict the impact of delay for several game and system configurations.

The rest of this paper is organized as follows: Section 2 presents work related to this paper; Section 3 describes the datasets and the procedure; Section 4 details the derivation of our models; Section 5 evaluates game performance using analytic models and simulation; Section 7 mentions some limitations of our approach; and Section 6 summarizes our conclusions and presents possible future work.

2 RELATED WORK

This section describes work related to the problem of modeling distribution times for the time needed to select moving targets with delay.

Models of User Input

Fitts' law is a seminal work in human-computer interaction and ergonomics that describes the time to select a stationary target based on the target distance and target width [6]. Fitts' law has been shown to be applicable to a variety of conditions (e.g., many different task settings [14]) and input devices (e.g., manual control devices as well as eye gaze [26]); it has also been extended to two dimensions [18], making it suitable for modeling target selection with a pointing device [23]; however, Fitts' aw by itself accounts for neither moving targets nor delay.

Jagacinski et al. [12], Hajri et al. [8], and Hoffmann [9] extended Fitts' law to moving targets, adding target speed to the model. Hoffman [10] revised Fitts' law to consider delay and Jota et al. [13] studied the impact of delay on target selection and dragging with touch devices,

Mackenzie and Ware [19] measured selection time and error rates when selecting stationary targets with delay. They found a pronounced multiplicative effect between delay and Fitts' Index of Difficulty (ID) [6] and proposed modification to Fitts' law that incorporates delay by including an additional term, $(c \cdot LAG)$, that is a weighting of delay plus the ID. Teather et al. [25] measured selection time for stationary targets of different sizes and delay and jitter, and confirmed similar findings to MacKenzie and Ware [19] for Fitts' law computations for ID. Friston et al. [7] confirmed earlier results of Fitts' law for low delay systesm and compared their model to those in both MacKenzie and Ware [19] and Teather et al. [25].

Pavlovych and Gutwin [20] measured accuracy and time in tracking and selecting fixed-sized targets that moved using Lissajous curves. Delay was only added to the the mouse click and not to the mouse movement. They found significant main effects for delay on the number of clicks and selection errors with a significant interaction between delay and target speed. Ivkovic et al. [11] measured selection time for fixed-sized, stationary targets and accuracy in tracking for two sizes of targets. They found significant main effects for delay on completion time and tracking and significant effects for task difficulty. The impact of delay on tracking was greater at lower target speeds than at higher target speeds. Long and Gutwin [16] measured selection time for different sized, moving targets. They found significant main effects for delay on selection time and accuracy and that the effects of delay are exacerbated by fast target speeds.

The models proposed provide for expected elapsed times and, in some cases, errors but do not model the distribution of elapsed times. The latter is needed for, as in the case of our work, simulations where player response times are selected from a range of possible values based on a model.

Game Actions

An overlapping area are approaches to studying the effects of delay on individual (aka *atomic*) game actions.

Claypool and Claypool [5] presented a general framework describing delay and game actions that includes *precision* – the accuracy required to complete the action successfully, and *deadline* – the time required to achieve the final outcome of the action. While helpful for explaining the effects of delay on games, the precision-deadline framework does not quantify the many possible parameters that make up game actions (e.g., dodge frequency).

Raaen and Eg [22] conducted experiments with a simple button and dial interface, letting users adjust delay based on their perceptions. They found users are capable of perceiving even low amounts of delay (around 66 milliseconds).

Long and Gutwin [15] studied the effects of delay on intercepting a moving target. They found target speed directly affects the impact of delay, with fast targets affected by delays as low as 50 ms but slower targets resilient to delays as high as 150 ms.

Pavloyvych and Stuerzlinger [21] and Pavloyvych and Gutwin [20] studied target selection and following for objects moving along Lissajous curves (smooth curves, with varying turns within the curve). They found tracking errors increase quickly for delays over 110 ms but the effects of target velocity on errors is close to linear.

In general, while these approaches have helped understand delay and fundamental game actions, they generally have not applied a model to the data gathered or if they have, as in the case of Long and Gutwin [15] the models are for average values and not distributions.

3 DATASETS

We use two sets of data obtained from prior user studies:[1] *Set-A* and *Set-B*. Each data set was obtained from users playing a game with controlled amounts of delay where the game

[1]Citations not provided to preserve anonymity during review.

focused on a single player's action – selecting a moving target with a mouse. Selecting a moving target is an action common to many PC game genres. Some examples include: 1) top-down shooters (e.g., *Nuclear Throne*, Vlambeer, 2015) where the player aims a projectile at opponents by moving the mouse to the intended target; 2) first person shooters (FPS) (e.g., *Call of Duty*, Activision, 2003) where players use the mouse to pan the game world to align a reticle over a moving opponent and shoot; and 3) multiplayer online battle arenas (MOBAs) (e.g., *League of Legends*, Riot Games, 2009) where players move a skill shot indicator with a mouse to target a moving opponent with a spell.

Game

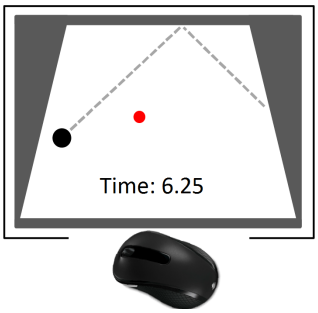

Figure 1: *Puck Hunt*. Users click on a moving target (the puck) with the mouse cursor (a red ball). The game adds delay to the mouse input and varies the target speed between rounds.

Both data sets are obtained from users playing a custom game called *Puck Hunt* that allows for the study of a moving target selection with controlled amounts of delay. In Puck Hunt, depicted in Figure 1, the user proceeds through a series of short rounds, where each round has a large black ball, the puck/target, that moves with kinematics, bouncing off the edges of the screen. The user moves the mouse to control the small red ball (a.k.a., the cursor) and attempts to select the target by moving the ball over the target and clicking the mouse button. Once the user has successfully selected the target, the target disappears and a notification pops up telling the user to prepare for the next round. Thereupon pressing any key, a new round starts, with the target at a new starting location with a new orientation and speed. The user is scored via a timer that counts up from zero at the beginning of each round, stopping when the target is selected. The target is a constant 28 mm.

In dataset Set-A, users select targets with three different speeds (150, 300 and 450 pixels/s) under 11 different delays (100, 125, 150, 175, 200, 225, 250, 275, 300, 400, and 500 ms), with each combination of delay and speed encountered 5 times.

In dataset Set-B, users select targets with three different speeds (550, 1100 and 1550 pixels/s) under 11 different delays (20, 45, 70, 95, 120, 145, 160, 195, 220, 320, and 420 ms), with each combination of delay and speed encountered 5 times.

For both Set-A and Set-B, users played Puck Hunt with a mouse.

Objective measures of performance recorded are the elapsed time to select the target and the number of clicks required to do so.

Procedure

All user studies were conducted in dedicated computer labs with computer hardware more than adequate to support the games and had LCD monitors.

For each study, participants first completed informed consent and demographic questionnaire forms before starting the game. The demographic questionnaire include the question "rate yourself as a computer gamer" with responses given on a 5 point scale (1 - low to 5 - high). The self-rating question was mandatory. The demographic questionnaire also included a gender question with options for "male", "female", "other" and "prefer not to say" – only one user did not specify either male or female.

Table 1: Summary of dataset variables

Dataset	Usrs	Gender	Rounds	Conditions
Set-A	51	43 ♂, 8 ♀	167	3 speeds, 11 delays
Set-B	32	23 ♂, 8 ♀, 1 ?	167	3 speeds, 11 delays
Combined	83	66 ♂, 16 ♀, 1 ?	334	6 speeds, 22 delays

Table 1 summarizes the major dataset variables, with the bottom row, "Combined", showing the users, gender and rounds of both datasets combined into one.

Table 2 shows the breakdown of self-rated skills for each dataset, with the mean and standard deviation reported by $\bar{x}$ and s in the last two columns, respectively. The bottom row shows the breakdown of both datasets combined into one. The datasets have a slight skew towards higher skills (mean skill slightly above 3 and mode 4 for each) but there are players of all skill levels in both sets.

4 MODELING SELECTION TIME

This section describes our methods used to process and analyze the user study data to derive models for the distribution of elapsed times to select moving targets with delay.

Preprocessing

In Puck Hunt, if the user's time to select the target surpasses 30 seconds, the round ends, and the elapsed time for that round is recorded as a 30. These 30 second values are not

Table 2: Dataset breakdown of self-rated skills

	Self-rated skill						
Dataset	1	2	3	4	5	$\bar{x}$	s
Set-A	1	3	5	24	18	4.1	0.9
Set-B	4	2	9	8	9	3.5	1.3
Combined	5	5	14	32	27	3.9	1.1

the elapsed times that would have been recorded if the trial continued, and so artificially impact any model of selection time that includes them. Thus, we look to replace values of exactly 30 seconds with estimates of the larger values they likely would have had if the round had continued (and the user had kept trying) until the target was selected.

In total, the game has 33 different combinations of speed and delay, called *difficulty levels*. For most difficulty levels, there are no elapsed times of 30 seconds. However, the higher difficulty levels (speeds above 500 px/s, and delay above 120 ms) have one or more 30 second values.

Table 3: Elapsed times above 30 seconds

Speed	Delay (ms)					
(px/s)	145	170	195	220	320	420
550	0 -	0 -	0 -	0 -	1 (0.4%)	*5* *(2.0%)*
1100	2 (0.8%)	1 (0.4%)	3 (1.2%)	2 (0.8%)	*18* *(7.1%)*	**56** **(22.0%)**
1550	3 (1.2%)	3 (1.2%)	*8* *(3.1%)*	**12** **(4.7%)**	**70** **(27.5%)**	**109** **(42.8%)**

Table 3 includes the number and percent of trials with elapsed times recorded as 30 seconds for high difficulty levels. All difficulties not in the table (i.e., easier levels) have no values recorded as 30. For the numbers with a normal font, only above 1% (or fewer) of the times are a 30, so we leave them as is.

For the difficulty levels in italics and bold, we estimate the elapsed times as if they were not artificially constrained, and generate synthetic points. Figure 2 depicts CDFs of the empirical data, along with previously derived models[2] at several example difficulty levels to illustrate our methods of generating elapsed times larger than 30. The x-axis is the elapsed time and the y-axis is the cumulative distribution. The solid lines are CDFs of empirical data, and the dashed lines with the same color are the corresponding models listed in Table 4. For the 4 difficulty levels with bold numbers in Table 3, we replace the original data with randomly generated points above 30 seconds using the models. For the 3 difficulty levels in italics, the corresponding CDF model does not surpass 30 seconds. For these difficulty levels, we model the tail of the distribution as a linear regression using the last 5 data points, and extend the CDF to 1 on the y-axis. We replace the original data with points evenly distributed on those extension lines.

[2]Citation not provided to preserve anonymity during review.

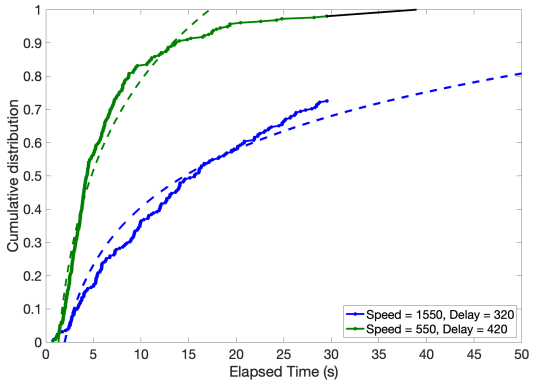

Figure 2: Example above 30 data

Table 4: Analytic models for CDFs for speed & delay combinations not including failures.

Speed	Delay	Equation	R^2	Mean (ms)
550	420	$0.39 \cdot ln(x) - 0.11$	0.96	6.2
1100	320	$0.34 \cdot ln(x) - 0.09$	0.98	8.2
1100	420	$0.27 \cdot ln(x) - 0.20$	0.94	23.1
1550	195	$0.29 \cdot ln(x) + 0.05$	1.00	7.4
1550	220	$0.30 \cdot ln(x) - 0.07$	0.97	10.3
1550	320	$0.25 \cdot ln(x) - 0.17$	0.94	30.0
1550	420	$0.21 \cdot ln(x) - 0.22$	0.93	66.8

Standardization

Before modeling, we standardized the mean and speed by subtracting the means and dividing by the standard deviations in Table 5.

Table 5: Values used for standardization

Factor	Mean	Std Dev
Delay (D)	206 ms	122
Speed (S)	683 px/s	488

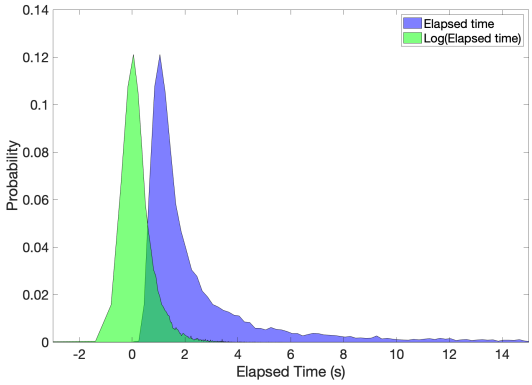

Figure 3: Probability distribution of elapsed time

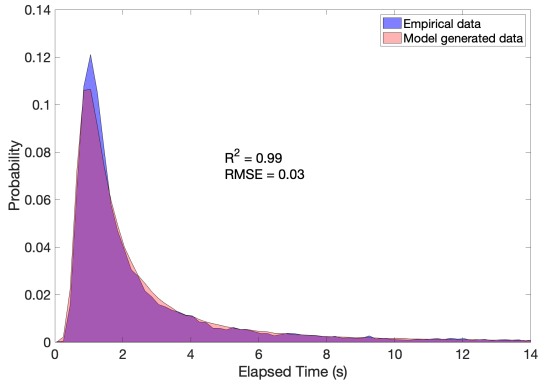

Figure 4: PDF of modeled data and empirical data

Normalization

Figure 3 shows the probability distribution of all elapsed times from the user studies. The x-axis is the elapsed time and the y-axis is the probability. The blue region, the distribution of all elapsed times, appears log-normal, which makes sense since human actions are impacted by many individual factors that, when put together, have an exponential distribution. We take the logarithm of the elapsed time to obtain a probability distribution in the green region that appears normal. The probability distribution of elapsed time at each difficulty level follows this pattern, too.

Modeling

We use multivariate, multiple regression to model the mean and standard deviation of the natural logarithm of the elapsed time. This will allow us to generate distributions of elapsed times for a given difficulty level, making it usable both for analytically modeling and for simulating game performance.

There are many possible models that fit the elapsed time data. We compare different regression models by using the coefficient of determination (R^2) as a measurement of how well observed outcomes are replicated by the model. In doing such a comparison, it might be tempting to choose the model with the maximum R^2, but this can over-fit the model to the data (i.e., the R^2 value can be increased by adding more terms). However, the adjusted R^2 modifies the R^2 based on the number of predictors in the model, increasing if the new term improves the model more than would be expected by chance and decreasing it otherwise. Overall, we want a parsimonious model – one that provides the desired prediction with as few terms as possible.

Table 6 summarizes the models explored. For the equations, the k parameters (e.g., k_1) are constants, e is the base of the natural logarithm (2.7182), d is the standardized delay and s is the standardized speed, where $d = \frac{D-206}{122}$ and $s = \frac{S-683}{488}$.

While model 10 has the highest R^2 for both mean and standard deviation, it is only slightly higher than model 8 while having significantly more terms (9 versus 4). Thus, we propose predicting mean and standard deviation for the log of the elapsed time:

$$\begin{aligned} mean(ln\ T) &= k_1 + k_2 d + k_3 s + k_4 ds \\ stddev(ln\ T) &= k_5 + k_6 d + k_7 s + k_8 ds \end{aligned} \tag{1}$$

where d and s are standardized from the delay (D) and target speed (S), respectively: k_1 through k_8 are constants derived from data gathered through the user studies.

Using our standardized user study data in Table 5 yields an adjusted R^2 for mean and standard deviation both at 0.96. The final model for the mean and standard deviation of the natural log of elapsed time is:

$$\begin{aligned} mean(ln\ T) &= 0.685 + 0.506 d + 0.605 s + 0.196 ds \\ stddev(ln\ T) &= 0.567 + 0.126 d + 0.225 s + 0.029 ds \end{aligned} \tag{2}$$

With the models predicting mean and standard deviation for ln T, given speed and delay, the normal distribution with the predicted mean and standard deviation can be used to generate a distribution of logarithmic elapsed times, taking the exponent to get the elapsed time. Figure 4 shows the model generated data compared to the actual data. The x-axis is the elapsed time in seconds, and the y-axis is the cumulative distribution. The blue shape is the probability distribution of the elapsed time for the empirical data, and the red shape is generated from the normal distribution using the modeled mean and standard deviation from Equation 2. Our model has excellent fit for the data with R^2 at 0.99 and RMSE at 0.03.

Table 6: Models predicting mean and standard deviation of selection time of a moving target with delay (d) and speed (s)

	Model	Mean R^2	Std Dev R^2
1.	$k_1 + k_2 s$	0.41	0.67
2.	$k_1 + k_2 e^s$	0.39	0.61
3.	$k_1 + k_2 d$	0.23	0.07
4.	$k_1 + k_2 d + k_3 s$	0.89	0.94
5.	$k_1 + k_2 e^d$	0.18	0.07
6.	$k_1 + k_2 e^d + k_3 e^s$	0.68	0.79
7.	$k_1 + k_2 d + k_3 d^2 + k_4 s + k_5 s^2$	0.88	0.94
8.	$\mathbf{k_1 + k_2 d + k_3 s + k_4 ds}$	**0.96**	**0.96**
9.	$k_1 + k_2 e^d + k_3 e^s + k_4 e^d e^s$	0.84	0.83
10.	$k_1 + k_2 d + k_3 d^2 + k_4 s + k_5 s^2 + k_6 ds + k_7 d^2 s + k_8 ds^2 + k_9 d^2 s^2$	0.97	0.96

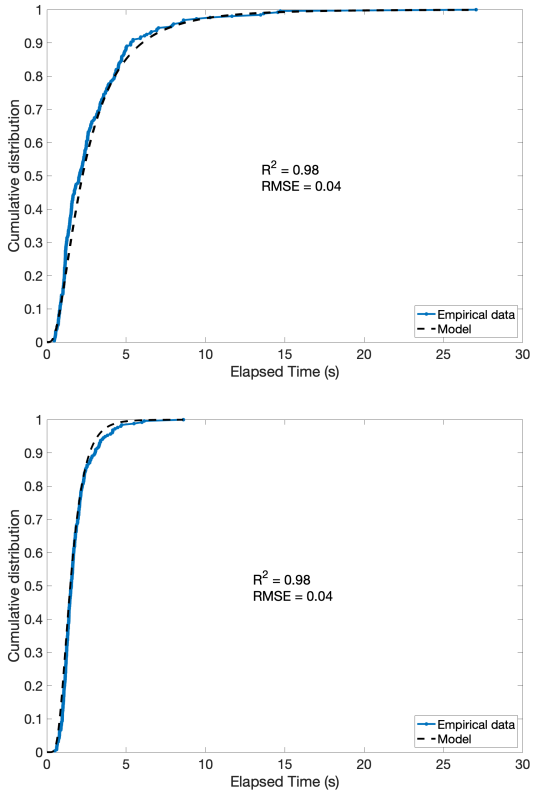

Figure 5: Example CDF data and models. Top: speed 1550 px/s and delay 70 ms. Bottom: speed 550 px/s and delay 170 ms

Figure 5 shows example CDFs of our model and data for speed at 1550 px/s and delay 70ms (top) and speed 550 px/s and delay 170 ms (bottom). The blue dots are the empirical elapsed times from the user study data and the dashed black lines the corresponding models at the respective difficulty. In all cases, the model fits the data well, with R^2 both at 0.98 and RMSE both at 0.04.

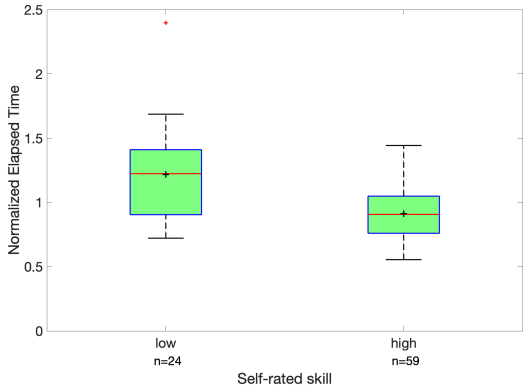

Figure 6: Combined skill groups

Player Skills

In the user studies, players rate their skills as computer gamers from 1 (low) to 5 (high). The mean self-rating was about 3.9, showing a slight skew to having "high ability". Based on our user sample, we divided users into low skill (24 users with self-rating 1-3) and high skill (59 users with self-rating 4-5).

The performance of each user is the average of their results across all trials in their user study (Set-A or Set-B). Since the games and tested conditions are slightly different between the two user studies, user results from one study cannot be directly compared (or combined) with the results from another. Hence, we normalize the data for each user study based on the average performance of all users in the same dataset. For example, since the average elapsed time to select a target across all users and all trials for the Set-B dataset is 1.6 seconds, each individual user in the Set-B dataset has their average elapsed time divided by 1.6. Users with normalized values below 1 are better than average and values above 1 are worse than average – e.g., a normalized score of 0.9 is

10% better than the average while a 2.0 is twice as bad as the average.

Figure 6 shows boxplots of normalized elapsed time on the y-axis for users clustered by skill group on the x-axis. Each box depicts quartiles and median with the mean shown with a '+'. Points higher or lower than 1.4 × the inter-quartile range are outliers, depicted by the dots. The whiskers span from the minimum non-outlier to the maximum non-outlier. The x-axis "n=" labels indicate the number of participants in each skill group.

From the figure, the mean and median of normalized elapsed times decrease (improve) with self-rated skill. The spread also indicates that some individuals with lower self-ratings perform better than users with higher self-ratings.

Since the elapsed time data was observed to be skewed right, comparisons of the two skill groups was done using a Mann-Whitney U test – a non-parametric test of the null hypothesis that it is equally likely that a randomly selected value from one group will be greater than or less than a randomly selected value from a second group. Effectively, this tests whether two independent self-rated skill group samples come from populations having the same distribution.

Table 7: Mann-Whitney U test for elapsed time by self-rated skill

Users		Median			
low	high	low	high	U	p value
24	59	1.22	0.91	321	**<.001**

Table 7 gives the results of the Mann-Whitney U tests. The "Users" and "Median" columns show the number of corresponding participants and median normalized elapsed times for the respective skill groups. The "U" and "p value" columns depict the test results. The test indicates that the difference in median elapsed of low skill compared to high skill is significant.

Modeling with Player Skills

We derive models for ln T parameterized by skill as we did for all users before, again finding the form of model 8 in Table 6 to be the most parsimonious.

The final models for the mean and standard deviation of ln T parameterized by player skill are shown in Table 8.

Figure 7 shows example CDF models for a 1100 px/s target and delay 320ms for a a low skill player (top) and a high skill player (bottom). For both graphs, the x-axis is the elapsed time and the y-axis is the cumulative distribution. The blue dots are the empirical elapsed times and the black dashed lines are the corresponding models. Our model fits both sets

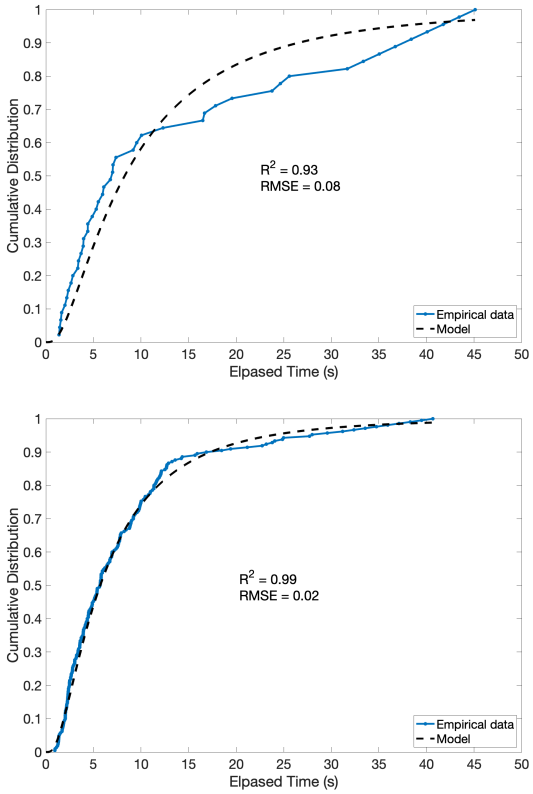

Figure 7: Example CDF for speed 1100 px/s and delay 320 ms. Top: low skill. Bottom: high skill

of data well, with R^2 at 0.93, RMSE at 0.08 for the low skill player, and R^2 at 0.99, RMSE at 0.02 for the high skill player.

Validation

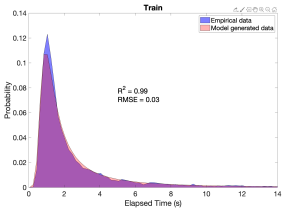

Figure 8: *Training*. Train the model with 80% of the data.

To validate the modeling approach, we randomly take 20% of data from each difficulty level(combination of target speed and delay) for validation, and train a model with the remaining 80% of data. Figure fig:train depicts how the resulting model fits the training data. The model fits the training data well with R^2 at 0.99 and RMSE at 0.03. Figure fig:validate depicts how the trained model fits the validation data. The model fits the validation data well with R^2 high at 0.99, and

Table 8: Models of moving target selection time with delay

Skill			Model	Adjusted R^2
All	$mean(ln\ T)$	=	$0.685 + 0.506d + 0.605s + 0.196ds$	0.96
	$stddev(ln.T)$	=	$0.567 + 0.126d + 0.225s + 0.029ds$	0.96
Low	$mean(ln\ T)$	=	$0.850 + 0.560d + 0.672s + 0.212ds$	0.96
	$stddev(ln.T)$	=	$0.589 + 0.118d + 0.253s + 0.009ds$	0.88
High	$mean(ln\ T)$	=	$0.605 + 0.468d + 0.625s + 0.208ds$	0.95
	$stddev(ln\ T)$	=	$0.539 + 0.109d + 0.227s + 0.041ds$	0.96

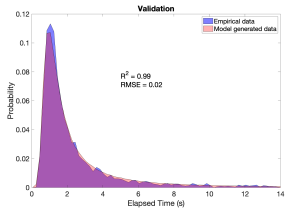

Figure 9: *Validation*. Validate trained model with 20% of the data.

RMSE low at 0.02. The modeling approach is validated as an effective and accurate approach to generate models that represent the distribution of moving target selection time. To make most of the data and to get better models, we use the models trained by all of the data as the analytic models for moving target selection.

5 EVALUATION

As a demonstration of the models use and to evaluate several specific game configurations, we use our model as the basis for analytically modeling player performance and for some game simulations. This allows us to explore the impact of game and system configuration on game performance over a range of delays.

Player Performance versus Delay

We begin by comparing the impact of delay on performance for target selection compared to reaction actions where a player responds immediately (e.g., by a key press or mouse click) to an event in the game. Using our model, selection time performance with delay is modeled analytically by $\frac{T(0)}{T(n)}$, where T(n) is the mean of elapsed time with delay at an average target speed at 450 px/s, calculated with our model. Reaction time actions are similarly modeled, but using the mathematical response time derivation by Ma et al. [17].

Figure 10 depicts the effects of delay on player performance for both actions. The x-axis is delay, in milliseconds, with a "0" representing the ideal case, and the y-axis the normalized performance with a "1" representing performance in the best case (0 delay). The blue line shows how player

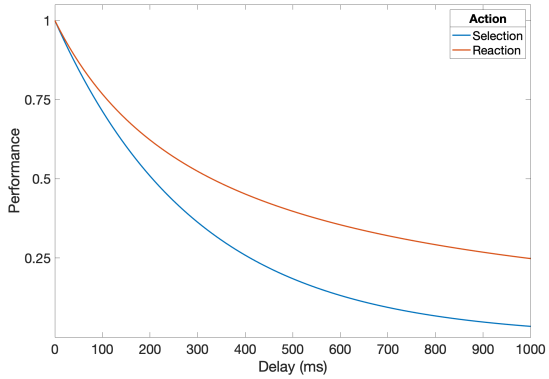

Figure 10: Player performance versus delay

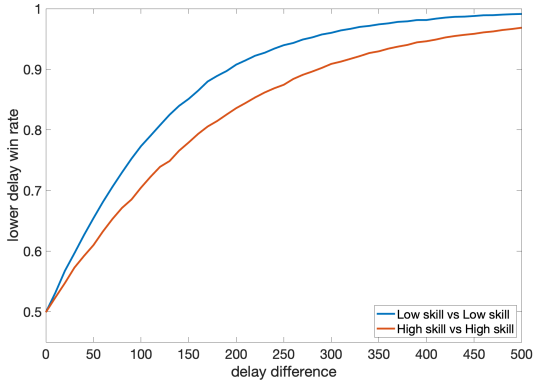

Figure 11: Win rate versus delay for different skill players

performance selecting a 450 px/s target decays with an increase in delay. The orange line shows how player reaction time performance decays with an increase in delay. From the figure, both actions have degraded by about 25% at a modest delay of 100 ms. The blue line has a steeper decreasing trend than the orange line, indicating that delay has more impact on selection actions than on reaction actions.

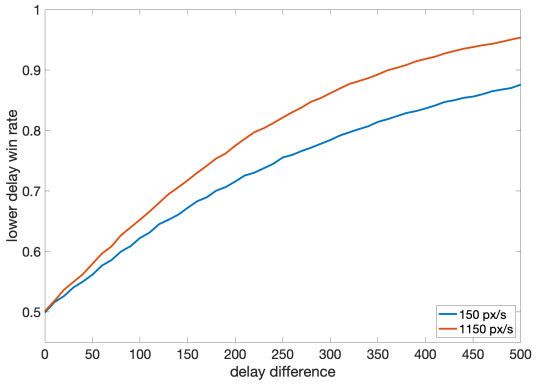

Figure 12: Win rate versus delay for difference target speeds

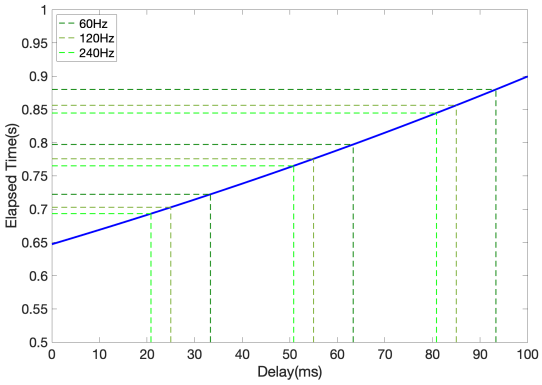

Figure 13: Frame rate and delay

Win Rate versus Delay – Skill

We simulate a shooter game where two players try to select (shoot) a target before their opponent. Both players have an equal base delay, but one player has extra network delay to evaluate the effects of unequal amounts of delay on matches with players of equal skill. We simulate 100k iterations of the game for each combination of added delay and player skill. Figure 11 depicts the results. The x-axis is the delay difference for the two players, and the y-axis is the win rate of the player with lower delay. The blue line is games with two low skill players, while the orange line is games with two high skill players. From the figure, even modest delay differences of 100 ms makes it about 50% harder for the delayed player to win. The blue line increases faster than the orange line, indicating delay impacts the games with higher skill players less.

Win Rate versus Delay – Target Speed

Figure 12 depicts how delay impacts win rate with different target speeds. This simulation is the same as before, but both players are of average skill and the target speed varies. The x-axis is the delay difference for the two players, and the y-axis is the win rate of the player with lower delay. The blue line is games with slow target speeds of 150 px/s, while the orange line is games with fast target speeds of 1150 px/s. The orange line increases faster than the blue line, indicating delay impacts player performance more for higher target speeds (i.e. harder games).

High Framerate Displays

We next simulate the potential benefits high framerates displays have on gameplay. We assume a configuration with a total delay of 55 ms and a 120 Hz display, and then evaluate the impact on performance for alternate setups using the same shooting game used above with target speeds of 300 px/s.

Figure 13 depicts the results. The blue line simulates an average skill player, the darkest green dashed line depicts perform if a 60 Hz display were used instead, and the lightest green dashed line if a 240 Hz display were used instead.

Table 9: Display frame rate

Delay(ms)	Hz	Elapsed Time(s)	Δ w/Hz	Δ w/Delay
	60	0.722	-2.70%	
25	120	0.703	—	9.41%
	240	0.693	1.42%	
	60	0.797	-2.71%	
55	120	0.776	—	—
	240	0.765	1.42%	
	60	0.880	-2.80%	
85	120	0.856	—	-10.31%
	240	0.845	1.29%	

Table 9 quantifies the results. Column #1 includes three total base delays, each with 120 Hz display. Column #2 shows the display frame rates simulated (60 Hz, 120 Hz, and 240 Hz). Column #3 has the simulated elapsed time at each base delay and Hz display. Column #4 provides the change in performance for the different displays from the base delay. And column #5 provides the change in performance for a 30 ms change in the base delay.

From the table and figure, the performance difference for a downgrade to the 60 Hz display is between -2.8% and -2.7%, and the performance difference for an upgrade to the 240 Hz is between 1.29% and 1.42%. Thus, an upgrade form 60Hz to 120Hz improves player performance more than an

upgrade from 120Hz to 240Hz. The performance difference from changing the base delay from 55ms to 25ms is 9.41%, and from 55ms to 85ms is -10.31%. In total, this suggests that base delay and network delay impacts player performance more than does the display frame rate. This confirms user study results measured by Spjut et al. [24].

6 CONCLUSION

With the growth in networking and cloud services, applications are increasingly hosted on servers, adding significant delay between user input and rendered results. This is true of computer games, where user input in response to game events can be delayed by the local system and the networking and processing by the server before rendered results are displayed on the screen. Understanding the impact of delay on game input can help build systems that better deal with the avoidable delays.

While user studies are effective for measuring the effects of delay on player performance, they are time intensive and, by necessity, typically have a limited range of game parameters they can test. Analytic models and simulations that mimic the behavior of players in games can complement user studies, providing for a broader range of study. This latter approach is most effective if the game modeled and simulated incorporates observed user behavior. This work leverages data gathered from two previous user studies to build a model that can be used for just such an approach – analytic models and simulations.

Eighty-three users playing 334 rounds of a game provided data on the time to select a moving target, the target moving with 6 different speeds and the mouse input subjected to 22 different amounts of delay. The resultant elapsed time data on the time to select the targets appears log-normal. We used multivariate, multiple regression to model the mean and standard deviation of natural log of the elapsed time, with additive linear terms for delay and speed and a multiplicative interaction term. This use of delay and speed is critical for accuracy of the models, as is the interaction term as there is an interplay of high delay and high speed that impacts elapsed times. The models can be used to generate a normal distribution of logarithmic elapsed times, then expanded by taking the exponential to get a distribution of elapsed times. Furthermore, the same approach is used to model elapsed time distributions based on self-rated user skill, which we ascertained could be differentiated into two tiers: low (self-rated skill 1-3) and high (self-rated skill 4-5). As proposed, our models have an excellent fit (R^2 around 0.98 and low root mean square error).

In addition to the main contribution of a model for the distribution of target selection times with delay, we demonstrate use of the model by analytically modeling and simulating player performance in a game that features target selection. The analytic modeling and simulation results show that for the evaluated game:

1 Even delays of only 100ms make it about 50% harder to win.
2 High skill players are less affected by delay than low skill players.
3 Delay affects players more for faster targets.
4 Improving delays by as little as 30ms can improve player performance more than increasing frame above 60 Hz.

7 LIMITATIONS

The model is limited by the details recorded and varied by the underlying user studies. As such, the model is only accurate over the range of target speeds and delays tested. For example, this means the model may not be accurate in extrapolating results to very low delays, such as might be encountered in future high end gaming systems, nor may it well-represent extremely small or extremely slow targets.

While target selection is a common action in both 2D and 3D games, the user studies were for a 2D game only so the model may not be accurate in 3D where perspective and target size may change with camera placement.

The self-rated skill is a coarse measure of player experience and a more detailed, multi-question self-assessment may provide for more accuracy in models of performance versus skill.

The studies including 66 males, but only 16 females participants and the vast majority are relatively young. Gamer demographics are much more evenly split across gender and more widely distributed by age.

While the models presented in this paper provide insights into a meaningful and fundamental measure of performance – the elapsed time to select – another important measure of performance for selection is accuracy. Accuracy in the datasets used in this work manifests itself in the number of clicks needed to select the target – any number greater than 1 is a "miss". Future work could model accuracy in a manner similar to the elapsed time models in this work. These accuracy models could be combined with our elapsed time models to simulate shooter games with ammunition, wherein accuracy matters in terms of number of shots taken or allowed.

The atomic action of moving target selection is only one of many fundamental game actions that are affected by delay. Future work could design and conduct user studies to gather data on the effects of delay on other atomic actions. Data from these studies could the modeled as in this paper, and those models combined with our models of target selection to simulate the effects of delay for a richer set of games. For example, a user study that assessed the impact of delay on

avatar navigation could provide data for models of the impact of delay on player avatar movement. These navigation models could be used in conjunction with our models of target selection to simulate a wide range of shooter games, allowing study of the effects of delay for the many commercial shooter games that have not been evaluated, nor possibly even built.

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 HCI – 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] De-Yu Chen and Hao-Tsung Yang dn Kuan-Ta Chen. 2013. Dude, the Source of Lags is on Your Computer. In *Proceedings of ACM Network and System Support for Games Workshop (NetGames)*. Denver, CO, USA.

[4] Kuan-Ta Chen, Yu-Chun Chang, Hwai-Jung Hsu, De-Yu Chen, Chun-Ying Huang, and Cheng-Hsin Hsu. 2014. On the Quality of Service of Cloud Gaming Systems. *IEEE Transactions on Multimedia* 26, 2 (Feb. 2014).

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

[6] 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.

[7] Sebastian Friston, Per Karlström, and Anthony Steed. 2016. The Effects of Low Latency on Pointing and Steering Tasks. *IEEE Transactions on Visualization and Computer Graphics* 22, 5 (May 2016), 1605–1615.

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

[9] Errol Hoffmann. 1991. Capture of Moving Targets: A Modification of Fitts' Law. *Ergonomics* 34, 2 (1991), 211–220.

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

[11] Zenja Ivkovic, Ian Stavness, Carl Gutwin, and Steven Sutcliffe. 2015. Quantifying and Mitigating the Negative Effects of Local Latencies on Aiming in 3D Shooter Games. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*. Seoul, Korea, 135–144.

[12] R. Jagacinski, D. Repperger, S. Ward, and M. Moran. 1980. A Test of Fitts' Law with Moving Targets. *The Journal of Human Factors and Ergonomics Society* 22, 2 (April 1980), 225–233.

[13] Ricardo Jota, Albert Ng, Paul Dietz, and Daniel Wigdor. 2013. How Fast is Fast Enough?: A Study of the Effects of Latency in Direct-touch Pointing Tasks. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*. Paris, France.

[14] R. Kerr. 1973. Movement Time in an Underwater Environment. *Journal of Motor Behavior* 5, 3 (1973), 175 – 178.

[15] M. Long and C. Gutwin. 2018. Characterizing and Modeling the Effects of Local Latency on Game Performance and Experience. In *Proceedings of the ACM Symposium on Computer-Human Interaction in Play (CHI Play)*. Melbourne, VC, Australia.

[16] Michael Long and Carl Gutwin. 2019. Effects of Local Latency on Game Pointing Devices and Game Pointing Tasks. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Glasgow, Scotland, UK.

[17] Tao Ma, John Holden, and R. A. Serota. 2013. Distribution of Human Response Times. *Complexity* 21, 6 (2013), 61–69.

[18] I. S. MacKenzie and W. Buxton. 1992. Extending Fitts' Law to Two-Dimensional Tasks. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*. Monterey, CA, USA, 219 – 226.

[19] I. Scott MacKenzie and Colin Ware. 1993. Lag As a Determinant of Human Performance in Interactive Systems. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Amsterdam, Netherlands). 6.

[20] A. Pavlovych and C. Gutwin. 2012. Assessing Target Acquisition and Tracking Performance for Complex Moving Targets in the Presence of Latency and Jitter. In *Proceedings of Graphics Interface*. Toronto, ON, Canada.

[21] A. Pavlovych and W. Stuerzlinger. 2011. Target Following Performance in the Presence of Latency, Jitter, and Signal Dropouts. In *Proceedings of Graphics Interface*. St. John's, NL, Canada, 33–40.

[22] K. Raaen and R. Eg. 2015. Instantaneous Human-Computer Interactions: Button Causes and Screen Effects. In *Proceedings of the 17th HCI International Conference*. Los Angeles, CA, USA.

[23] R. W. Soukoreff and I. S. MacKenzie. 2004. Towards a Standard for Pointing Device Evaluation – Perspectives on 27 Years of Fitts' Law Research in HCI. *Elsevier International Journal of Human-Computer Studies* 61, 6 (2004), 751 – 789.

[24] Josef Spjut, Ben Boudaoud, Kamran Binaee, Jonghyun Kim, Alexander Majercik, Morgan McGuire, David Luebke, and Joohwan Kim. 2019. Latency of 30 ms Benefits First Person Targeting Tasks More Than Refresh Rate Above 60 Hz. In *SIGGRAPH Asia 2019 Technical Briefs*. 110–113.

[25] R. J. Teather, A. Pavlovych, W. Stuerzlinger, and I. S. MacKenzie. 2009. Effects of Tracking Technology, Latency, and Spatial Jitter on Object Movement. In *Proceedings of the IEEE Symposium on 3D User Interfaces*. Lafayette, LA, USA, 43–50.

[26] C. Ware and H. Mikaelian. 1987. An Evaluation of an Eye Tracker as a Device for Computer Input. In *Proceedings of ACM Human Factors in Comp. Systems and Graphics Interface*. Toronto, Canada.