The Impact of Latency on Navigation in a First-Person Perspective Game

Shengmei Liu, Mark Claypool
sliu7,claypool@wpi.edu
Worcester Polytechnic Institute
Worcester, Massachusetts, USA

ABSTRACT

Competitive first-person shooter games are played over a network, where latency can degrade player performance. To better understand latency's impact, a promising approach is to study how latency affects individual game actions, such as moving and shooting. While target selection (aiming and shooting at an opponent) is fairly well studied, navigation (moving an avatar into position) is not. This paper presents results from a 30-person user study that evaluates the impact of latency on first-person navigation using a custom "hide and seek" game that isolates avatar movement in a manner intended to be similar to movement in a first-person shooter game. Analysis of the results shows latency has pronounced effects on player performance (score and seek positioning), with subjective opinions on Quality of Experience following suit.

CCS CONCEPTS

• **Applied computing** → **Computer games**; • **Human-centered computing** → *User studies.*

KEYWORDS

skill, gamer, FPS, user study, navigation, lag

ACM Reference Format:
Shengmei Liu, Mark Claypool. 2022. The Impact of Latency on Navigation in a First-Person Perspective Game. In *CHI Conference on Human Factors in Computing Systems (CHI '22), April 29-May 5, 2022, New Orleans, LA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3491102.3517660

1 INTRODUCTION

Computer games are one of the world's most popular forms of entertainment, with global sales increasing at an annual rate of 10% or more [46]. The largest esports prize pools are about $25 million USD [15], larger even than traditional sports, and have prize pools that range from about $2 to $20 million USD [35]. By 2023, there are expected to be about 300 million frequent viewers of esports worldwide, an increase from 173 million in 2018 [19].

Among the myriad of game genres available and played with esports, the first-person shooter game is one of the most popular. In first-person shooter games, players take a first-person perspective of an avatar and then move and shoot targets to accomplish game goals, often playing with other people as teammates or opponents. Latency – delay between a player's input and the game responding with audible or visual output – makes first-person shooter games less responsive, degrading player performance and hurting quality of experience. There are two main sources of latency in first-person shooter games: 1) from the local system, such as from the mouse, OS and monitor, and 2) from the network between the client and the server. While both sources of latency affect the player, they impact the game differently -– local latency lags all player input until game output, while network latency lags communication with the server. This means local latency makes game controls feel unresponsive, while network latency makes player actions resolved later by the server.

There have been numerous studies on latency and commercial games [18, 20], especially network latency and first-person shooter games [3–5, 24, 28, 30, 40] owing to the sensitivity of first-person shooter games to network latency and the popularity of first-person shooter games in the competitive and esports scenes. The most common approach to assess latency is via user studies, but these can be expensive and time-consuming. Moreover, given the wide-variety of games even within a single genre (e.g., consider the scope of first-person shooter games available today) it is not practical and perhaps not even possible to cover all current and future game configurations with user studies.

An alternate approach is to study the effects of latency on individual game actions [32, 41] which has the potential to generalize to many games and even other interactive applications. For example, studies of target selection – positioning a pointer on top of an object and pushing a button – may yield results useful for understanding any game that has target selection as a game action. Studies assessing the impact of latency on target selection have varied target parameters (e.g., target size and distance) [16, 21, 34, 45] and target motion (e.g., velocity and acceleration) [12, 13, 38, 39], as well as pointing device [11, 33]. A key outcome of some of these approaches are analytic models that can explain and predict the effects of latency for a wide-range of games and latency conditions.

In regards to first-person shooter games, target selection has many similarities to aiming and shooting. However, the other first-person shooter action that has received little attention is navigation. Navigation in a first-person shooter game is when a player re-positions their avatar in order to shoot an opponent or to avoid being shot. On a PC, first-person navigation is typically done using the mouse to change direction by re-orienting the avatar and the WASD keys on the keyboard to move the avatar forward, left, right and back, respectively. While studies of steering [1, 22, 47] may have a bearing on navigation, most do not assess user performance

CHI '22, April 29-May 5, 2022, New Orleans, LA, USA

ACM ISBN 978-1-4503-9157-3/22/04…$15.00
https://doi.org/10.1145/3491102.3517660

with latency and those that do [17], do not include the adversarial nature of opponents in first-person shooter combat, as well as steer only with a mouse and not the keyboard. Known work in navigation in first-person shooter games using a keyboard [10] has not considered the effects of latency or competition with other players, nor provided models for general use. A better understanding of how latency affects navigation in a first-person game can provide insight into the effects of latency on first-person shooter games, and a model of first-person navigation can be useful for generalizing latency's affects.

The rest of this paper is organized as follows: Section 2 describes previous work on latency and games related to this paper; Section 3 details our methodology, including game and user study design and execution; Sections 4 and 5 provide the results and analysis, respectively, from the user study; Section 6 discusses limitations of our approach, including future work; and Section 7 summarizes our conclusions.

2 RELATED WORK

This section describes related work in three main areas: navigation in games (Section 2.1), local latency and games (Section 2.2) and network latency and games (Section 2.3).

2.1 Navigation in Games

Navigation in games is when a player moves the position of an avatar from one area in a virtual world to another. Often, a player wants to navigate as quickly as possible without colliding with obstacles in the game world.

Accot and Zhai [1] propose a *steering law* that predicts the total time required for a user to guide a pointing device (e.g., a mouse) from a starting location, through a tunnel on the screen (e.g., a set of nested menus), to an ending location (e.g., a menu option) while staying within the tunnel confines. Based on the law, the steering time is a function of the path width. The same authors [2] evaluate the model with 5 different input devices on 2 steering tasks and find the law holds for all devices tested, with greater than 0.98 correlation. Kattinakere et al. [22] enhance the steering law with conditions for both constrained and unconstrained paths. They propose a model and conduct 4 experiments on goal passing and tunneling tasks, demonstrating that the model can be used to effectively predict movement times when steering through constrained paths in above-the-surface interaction layers. Zhai et al. [47] evaluate the steering law in virtual reality (VR), having users do a VR locomotion task with a controlled path width and shape. They find that the steering law holds for VR navigation, as well. Friston et al. [17] assess the steering law with latency and find that latencies as low as 16 milliseconds degrade user performance and that this effect is non-linear.

This related research shows the steering law can accurately predict performance for a wide range of devices and in virtual worlds, with user performance degrading with latency. Our work complements this work by examining first-person navigation in a computer game with latency, but where our users' navigation is not constrained with a fixed path and where navigation is via mouse for orientation and keyboard for locomotion.

For related work in computer games, Pantel and Wolf [37] measure the time to steer a car around a race track when input is delayed by added latency. Their experiments suggest 50 milliseconds of latency is acceptable for users, but that the time to complete a lap of the track increases sharply with latency – roughly doubling for 250 milliseconds of latency. Claypool and Claypool [10] study the impact on latency on navigation and shooting in the commercial first-person shooter game *Quake 3* (id Software, 1999) with different frame rates and resolutions. They find resolution has little impact on both tasks, but frame rate has a pronounced effect on performance and quality of experience, more so for shooting than for navigation. Sabet et al. [42] investigate player quality of experience (QoE) for navigating a car around a track for the commercial game *Need for Speed Shift 2 - Unleashed* (Electronic Arts, 2011) as it relates to performance with latency. They find latency degrades player performance with latency which, in turn, lowers the quality of experience.

This related work in computer games shows player navigation is affected by latency (and other system conditions). Our study confirms these effects (i.e., latency degrades player performance and QoE), while providing measurements specific to navigation in a first-person game.

Navigation in virtual reality (VR), a kin of many computer games, has been widely studied. Boletsis and Cedergren [7] compare user experience on contemporary and prevalent VR locomotion techniques and find walking-in-place to be the most immersive, a controller/joystick to be easiest to use, and teleportation to be most effective. Chardonnet et al. [9] evaluate the influence of virtual barrier distance and navigation interface on cybersickness, and suggest combinations that enhance user experience. Serge and Moss [44] investigate cybersickness with a head-mounted display and find that sickness is relevant even with the minimal latencies measured. Buker et al. [8] study the effect of head movement frequency and predictive compensation with a helmet-mounted display. They find that apparent latency increased significantly as head movement frequency increased, and that latency compensation not only reduces latency but also sickness.

This research has devised effective VR navigation techniques and showed cybersickness' can occur in VR even with low latencies. Our research is related in our study of latency and navigation, but we use the dominant navigation technique for first-person computer games on a PC – a keyboard for locomotion and a mouse for direction – with our focus on performance not cybersickness, the latter being less common outside of VR.

2.2 Local Latency and Games

Understanding the effects of local latency on games and game-like actions can help motivate the design and development of end-host systems that benefit game players and other users doing interactive tasks.

Claypool et al. [13] measure player performance for target selection in a simple 2d game, finding target selection time increases exponentially with local latency and target speed. Ivkovic et al. [21] find significant effects for local latency on target tracking and acquisition tasks, both with and without aim assistance, and with a greater effect for higher target speeds. Long and Gutwin [32]

find target speed directly affects target acquisition with latency, with fast targets affected by latencies as low as 50 milliseconds but slower targets resilient to latencies as high as 150 milliseconds. Long and Gutwin [33] compare the effects of local latency across 4 different gaming devices, demonstrating that latency affects each device differently for moving target selection. Investigating using a full game, Liu et al. [30] measure performance and quality of experienced for skilled players in *Counter Strike: Global Offensive* (Valve, 2012) showing both degrade linearly with local latency – both subjective and objective scores decrease about 20% with a 100 millisecond increase in local latency.

While useful for understanding and even modeling the effects of local latencies on some gaming tasks and even a first-person game, the impact of local latency on navigation has not been assessed, particularly as it relates to first-person navigation in games. Our study uses some of the same latency values from this previous work to allow for comparison of results. Similarly, our results for navigation can be directly compared to previous results for target selection.

2.3 Network Latency and Games

Understanding the effects of network latency on games can help motivate deployment or development of algorithms that mitigate latency to improve user experience when interacting with other players over a network. This section concentrates on first-person games.

Bernier [6] describes latency compensation techniques that favor the player shooting over the player moving to avoid being shot. Specifically, the *lag compensation* technique resolves firing actions at the time the shooter carried them out at the expense of more recent movement actions by an opponent. Our results reinforce this by showing navigation for a player avoiding being seen (hence, shot) is less affected by latency than is navigation by a player seeking an opponent (e.g., for shooting).

Dick et al. [14] show via a survey that players generally think about 120 milliseconds is the maximum tolerable latency for a network game, regardless of game genre, but their user study shows players find 150 milliseconds acceptable for the two first-person shooter games tested. Amin et al. [3] find player experience defines and determines the sensitivity to latency for the first-person shooter game *Call of Duty* (Activision, 2003), with competitive gamers more adept at compensating for impaired conditions. Armitage et al. [4] estimate the latency tolerance threshold for the first-person shooter game *Quake 3* (id, 1999) to be about 150-180 milliseconds. Quax et al. [40] show for the first-person shooter game *Unreal Tournament 2003* (Epic, 2003) that latency and latency jitter under 100 milliseconds can degrade player performance and quality of experience. Holfeld et al. [20] find players of the casual game *Minecraft* (Mojang, 2011) are insensitive to network latencies of up to 1 second. Liu et al. [28, 29] show that while local latency in the first-person shooter game *CS:GO* (Valve, 2012) has a higher impact on players, reductions in network latency also benefit player performance and quality of experience.

This previous research shows first-person shooter games can be affected by network latencies as low as 100 milliseconds, but not necessarily all first-person games (e.g., Minecraft) are so affected. Our work builds upon this research by focusing on a first-person game with a single action – navigation – while using similar latency values (e.g., 0 to 200 milliseconds of network latency) to allow for comparison with previous results.

3 METHODOLOGY

In order to assess the effects of latency on navigation in a first-person game, we built a custom game that isolates the navigation action, added controlled amounts of local and network latency, recruited participants for a user study, and measured player performance and quality of experience.

3.1 Hide and Seek Game

We designed and implemented a custom first-person game in Unity that isolated the action of first-person navigation in an environment akin to a first-person shooter. Our game is a two-player game called *Hide and Seek*, shown via screen shot in Figure 2. At any given time, one player is the hider and the other player is the seeker. The goal for each player is the spot the opponent's avatar when the seeker, and hide from the opponent's avatar when the hider. These roles are intended to represent typical interactions in first-person shooter games where a player navigates to get an opponent in sight to shoot at, and, similarly, navigates to hide from an opponent while being shot at. In our game, the roles switch every 2-6 seconds – this abrupt and random switching of roles means to capture the dynamics in a first-person shooter game where a player is both hunting (trying to shoot an opponent) and hunted (trying to avoid being shot) in a short amount of time. Anecdotally, several users said, unprompted, that the game tension felt like a first-person shooter game, albeit without the weapons.

The update rate for the game engine is fixed at 50 frames per second. For every frame, if the seeker can see any part of the hider, the seeker earns a point; otherwise, the hider earns a point. A game round terminates after 40 seconds. While the roles are switched randomly, we ensure each player is the hider for exactly half the time (i.e., 20 seconds per round) and the seeker for half the time. When the timer runs out, the player with more points wins. Each frame, the game logs whether the hider or the seeker gets points, the running score for both players, and the keyboard and mouse actions.

Hide and Seek has one map, depicted in Figure 1. The map is a single, square room, 36 meters in length and width, with multiple obstacles to mimic maps in typical first-person shooter games where terrain can play a role in the combat. The player's avatar spawns at a random location on the map near, but not currently in view of, the opposing player. Upon spawning, the game provides a countdown timer for each player until the round starts. Figure 2 shows a Hide and Seek screenshot where the player is currently a seeker. Semi-transparent green "Seek!" or red "Hide!" messages in the middle of the screen inform the player of their current role. The round score and countdown timer are shown in blue at the top of the screen. In the screenshot, the opponent is in sight at that moment, thus the player (a seeker) is gaining points as long as the opponent remains visible (or the roles switch).

Hide and Seek has a client-server architecture typical of most multi-player network games where the authoritative server keeps

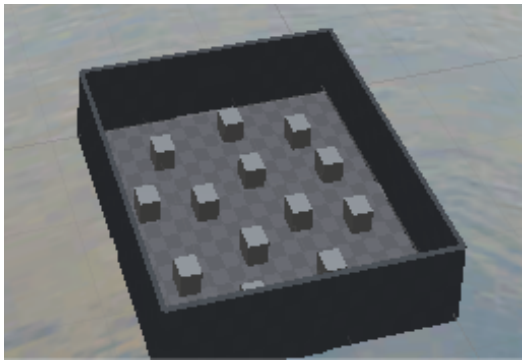

Figure 1: Overhead view of the Hide and Seek map. The map is a 36x36 meter room with 2.5 meter gray cubes as obstacles.

Figure 2: Hide and Seek screenshot. The green "Seek!" text informs the player their current role is a seeker. The score and countdown timer are in blue at the top of the screen. The opponent is in sight in this screenshot, thus the player gains points at this moment.

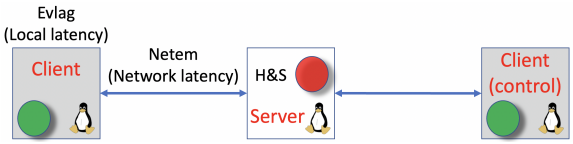

Figure 3: Hide and Seek computer configuration. Server and clients run Linux and are connected by a high-bitrate LAN. All computers run Hide and Seek, but the server has the authoritative game state. Network latency is added with Linux's tc with Netem and local latency is added with EvLag.

the master world state and communicates state updates to the clients.

3.2 Testbed Setup

We setup the game for our user study in a dedicated, on-campus computer lab. The testbed setup is depicted in Figure 3. The server hosts the game and is connected via high-speed LAN to the clients. The clients and server are Alienware PCs with Intel i7-4790K CPUs @4 GHz with 16 GB RAM and an Intel HD 4600 graphics card. The clients are each equipped with a gaming mouse and monitor so as to minimize local system latency and maintain consistency. The clients have a 25" Lenovo Legion monitor running at 1920x1080 pixels displayed at 16:9 and 240 Hz, with AMD FreeSync and a 1 ms response time. The mouse is a Logitech G502 12k DPI with a 1000 Hz polling rate. The clients and the server run Ubuntu 20.04 LTS, with Linux kernel version 5.4.

The local system latency was measured with a 1000 frame/s camera (a Casio EX-ZR100) setup to capture the moment a user presses the mouse button and the resulting screen output. By manually examining the video frames, the frame time when the mouse is clicked is subtracted from the frame time the result is visible, giving the local latency. This measurement method was done 10 times on our client, yielding an average base latency of 22 milliseconds with a standard deviation of 5 milliseconds.

Table 1: Latency values for the user study.

Latency Type	Values (ms)
Local	25, 100, 175
Network	0, 100, 200

Local latency delays all input until resulting rendered output, whereas network latency delays receipt of the player's action at the server and subsequent server response to the client. Since the Hide and Seek server is authoritative, the client cannot update the position of an avatar until the server response has arrived. Thus, for Hide and Seek (as for all client-server games without latency compensation), local latency manifests similarly to network latency. Player movement input until resulting avatar movement is seen on the screen is delayed by at least the sum of the local latency and the network latency.

Our intent is to assess local latencies over ranges that might typically be found in personal computers, which range from about 25 milliseconds for a fast gaming system, are around 100 milliseconds for a typical computer system, and can be 175 milliseconds for a slower gaming system [21]. We added latency to all mouse and keyboard input using EvLag [25] – an open-source tool for Linux that adds a constant amount of latency to any input device. Given our client has an average local latency of 22 milliseconds, EvLag adds either 3, 78 or 153 milliseconds of latency for resulting total local latencies of 25, 100 and 175 milliseconds, respectively, as shown in Table 1. Note that 25, 100 and 175 milliseconds are average values since the underlying system does not have a fixed, constant latency, consistent with all personal computers that do not have real-time control over devices, operating system scheduling and game computations.

Similarly, our intent is to assess network latencies over ranges typically experienced by PC network game players, which can be near 0 milliseconds for a local area network (LAN) game, 100 milliseconds for a reasonable Internet connection, and 200 milliseconds for a slower Internet connection [36]. We added network latency to the server uplink and downlink equally using Linux tc with Netem[1] – a network control tool. The total network latency added to the client was either of 0, 100, or 200 milliseconds as indicated in Table 1.

3.3 User Study Procedure

Before the launch of the formal user study, a pilot study with 3 volunteers was conducted in order to test the viability of the procedure and tune the study settings. The pilot study results helped adjust round length, map size and layout, number of rounds, latency values and user instructions.

The IRB-approved user study was conducted during the COVID pandemic, so everyone wore masks and respected social distancing requirements. Before starting each study, all computer devices and touched surfaces were carefully sanitized. Interested participants first filled out a demographics questionnaire with questions on game-related experience. Selected users were invited to the lab at a pre-set time, where they began by signing a consent form and positioning themselves comfortably at the test client computer.

Users first did a custom reaction-time test written in Javascript and launched via a Chrome Web browser. In the test, users waited for a screen color change then clicked the mouse as quickly as possible, doing this 10 times. The average of the 10 values provides a measure of user reaction time.

For consistency of play conditions, all participants played against the same opponent, who served as a control. We considered using a computer-controlled, AI opponent (a "bot") for both consistency and, once our game is public, reproducibility of the results. Unfortunately, even in modern, commercial first-person shooter games bots often behave quite differently than do human opponents, and by far the most popular first-person shooter game modes are played are human-versus-human. For our Hide and Seek game, we were not certain of our ability to create a bot that used the same tactics and strategy as would a human Hide and Seek player. Since realistic opponent behavior was deemed essential, not necessarily for score (which might vary depending upon the skill of an opponent), but to assess the relative effects of latency on performance and experience, we used the same human opponent for all users. Network latency and local latency were only applied to the participant's avatar and not to the control avatar, as indicated in Figure 3.

Users started by playing one practice round without any added latency to get familiar with the game. This data was not analyzed. Users next played additional rounds, each with a different local latency (25, 100, or 175 milliseconds) and network latency (0, 100, or 200 milliseconds), randomly shuffled. Each combination of local latency and network latency was repeated 3 times, for a total 27 rounds (plus the practice round) for each user. After each round, users provided a subjective Mean Opinion Score (MOS) on a discrete 5-point Likert scale about the game experience in the preceding round. The question was "Rate the quality of the previous game round", and players chose an answer from 5 options: Excellent, Good, Fair, Poor or Bad. Since users played a lot of game rounds (27) and each round was short (40 seconds), we only asked the one question to avoid fatigue. Plus, our previous work assessing user experience for first-person shooter games with short rounds found strong correlations between this single question and additional questions that could be of interest (e.g., immersion, ease of control) [30]. After completing the survey, the next round would commence when the user was ready, but users could take as much time as needed before starting the subsequent round,

[1]https://wiki.linuxfoundation.org/networking/netem

Table 2: Participant demographics

Users	Age (yrs)	Gender	Gaming per week (hours)	Game Self-rating	FPS Self-rating	Reaction-time (ms)
30	23.1 (4.0)	26 ♂4 ♀	10.4 (8.3)	3.4 (1.1)	3.1 (1.0)	227.2 (40.0)

It took each user about 30 minutes to complete all the tasks in the study. A user study proctor was available for questions and trouble-shooting for the duration.

After completing all the game rounds, users were given a questionnaire with additional demographics questions about gamer experience – average time spent playing games and self-rated expertise with computer games.

In summary, the procedure each user followed was:

(1) Fill out the screener questionnaire to ensure interest in participation and computer use.
(2) Come to the dedicated lab with pre-configured computers and sign the consent form.
(3) Adjust the computer chair height and monitor angle and height so as to be comfortably looking at the center of the screen.
(4) Read the instructions regarding setup and game controls on the desktop.
(5) Complete the reaction-time test. (Takes about 30 seconds.)
(6) Complete the hide and seek game rounds (1 practice round and 27 rounds with shuffled latencies), including the QoE surveys after each round. Take breaks between rounds, if needed. (Takes a bit less than 30 minutes, total.)
(7) Complete the final demographics questionnaire.

All users were eligible for a raffle to win a $25 USD Amazon gift card upon completion of the study, and many users received playtesting credit for relevant classes in which they were enrolled.

Study participants were solicited via university email lists.

4 RESULTS

This section provides a demographics for the participants (Section 4.1) followed by a summary of the results (Section 4.2).

4.1 Demographics

Table 2 summarizes the demographic information for the user study participants. First-person shooter (FPS) self-rating is on a five-point scale, 1-low to 5-high. For age, FPS self-rating, and reaction times, the mean values are given with standard deviations in parentheses. Our user study had 30 participants, ranging from 18-31 years old but with the large majority of typical college age. Gender breakdown is

predominantly male (26 males versus 4 females). We were slightly disappointed by the low number of female participants, but note that this reflects the gender breakdown of first-person shooter game players (about 7% of first-person shooter gamers are women [23]) and our sample pool of students at our university skews male. Half of the participants played 10 or more hours of computer games per week. User self-ratings in general computer games slightly skews towards above the mid-point (mean 3.4), with self-rating in FPS games slightly lower (mean 3.1). Most participants majored in Robotics Engineering, Computer Science, or Game Development.

4.2 Summary

Table 3 summarizes the study results in tabular form, providing a linear regression for the mean values with latency (slope and y-intercept), adjusted coefficient of determination (R^2) and significance (*pvalue*). Statistical significance at an alpha of 0.05 is indicated in bold. Overall, the results are significant in nearly all cases: player score, quality of experience (QoE), seeker time with the opponent in sight, and actions per minute. The notable exception is for the hider time avoiding the opponent where the result is not significant. In other words, latency does not significantly change a player's ability to hide from an opponent.

Table 3: Results Summary

metric	slope	y-intercept	R^2	p value
score	-0.029	49.3	0.93	**<.001**
QoE	-0.005	4.03	0.95	**<.001**
seeker time	-0.010	7.38	0.96	**<.001**
hider time	-0.001	12.23	0.34	0.10
actions / minute	-0.060	92.37	0.95	**<.001**

5 ANALYSIS

This section first analyzes the consistency of the human opponent as a scrutiny of the consistency of play conditions across players (Section 5.1), then compares the effects of local latency and network latency (Section 5.2), followed by the core results – user performance (Section 5.3) and Quality of Experience (Section 5.4) in the presence of latency. Additional analysis examines the total time with the opponent in sight as a seeker and out of the opponent's sight as a hider (Section 5.5) and actions per minute (Section 5.6) with latency. Lastly, analytical models generalize the individual time intervals for hiding and seeking in our game (Section 5.7).

5.1 Opponent Performance

In using the same human opponent, there is a risk that the opponent either: a) gets better at the game over time, making the game more difficult for later participants, or b) gets fatigued and plays worse over time, making the game easier for later participants. Figure 4 depicts the performance of the opponent across the 30 users. The x-axis is the participant (player) number from one to thirty by participation order, and the y-axis is the score percent of the opponent – a score above 50 means the opponent got more than half the points and won, while a score below 50 means the participant got more than half the points and won. The circles are the mean scores of all game rounds against the opponent, and the dashed line is a linear regression through the mean values.

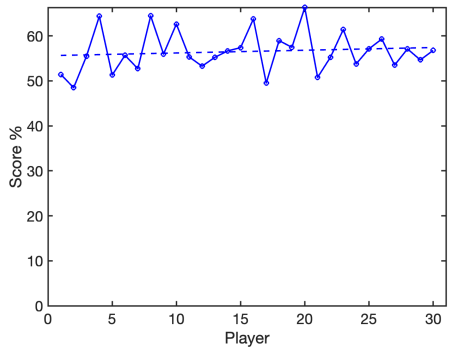

Figure 4: Opponent score versus player order.

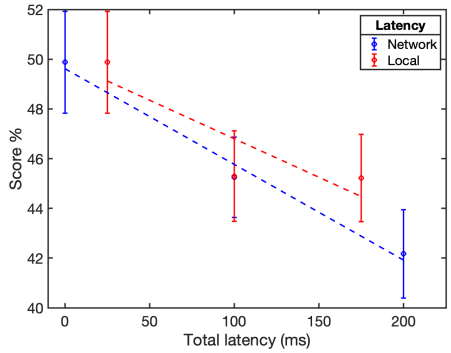

Figure 5: Score versus latency – comparing local latency and network latency.

From the graph, there is visible variation in performance across participants, and while the opponent won more often than lost (the opponent never had added latency, only the participant), some participants beat the opponent. The p value of the linear regression is 0.54, indicating that there is no statistically significant difference in the opponent's performance across the 30 players. Correspondingly, the regression line is visually flat, suggesting the opponent had consistent effort and skill over time. In other words, the participants likely faced a similar challenge regardless of their participation order.

Note, all analysis is done using the participant's data, not the opponent's data.

5.2 Local Latency versus Network Latency

Previous work has shown that local latency has a higher impact on player performance than network latency [29], but that was for a commercial game that has built-in latency compensation techniques. Our study allows comparison of player performance with local latency versus network latency in the absence of latency compensation.

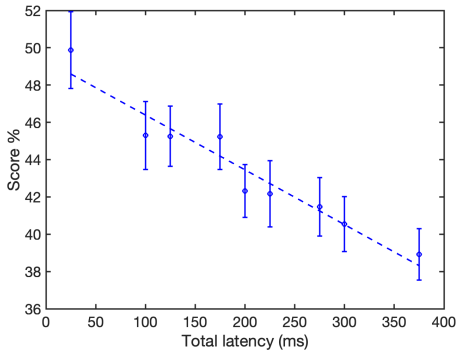

Figure 6: Score versus latency.

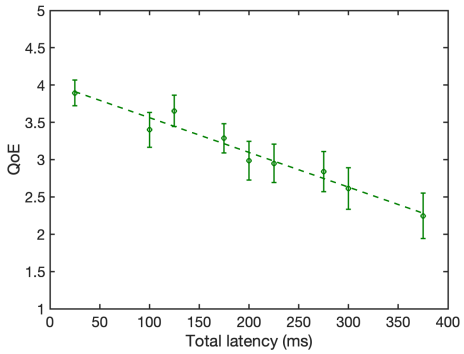

Figure 7: Quality of Experience versus latency.

We measure player performance based on the points earned. For each frame (50 f/s), if the player is a seeker and has the opponent in sight or if the player is hider and is not visible by the opponent, the player earns a point. For easier analysis, we convert the points to a score percentage which is the number of points earned divided by the maximum possible number of points (2000), multiplied by 100. A score percent above 50 means the participant won, while a score percent below 50 means the opponent won.

Figure 5 depicts score percent versus latency for network latency and local latency. The x-axis is the total latency (network plus local) in milliseconds. The y-axis is score percent for the player. The circles are the score percent means bounded by 95% confidence intervals and the dashed lines are linear regressions through the mean values. Blue is for rounds with network latency only, but without any added local latency and red is for rounds with local latency only without extra network latency. From the graph, player performance degrades with both types of latency. While the slope for the network latency regression appears slightly steeper than the slope for the local latency regression, a regression for a unified model with latency type (local/network) as a parameter shows the latency type parameter is not statistically significant ($p = 0.36$). Results of a two-sample Kolmogorov-Smirnov (KS) test provide a KS statistic (D) value of 0.08 and p value of 0.29 and point to the null hypothesis being accepted, suggesting that the two samples are drawn from the same distribution at the 5% significance level. Given this, and since our expectation is that in the absence of latency compensation local latency and network latency impact navigation identically, all subsequent analysis does not differentiate the data by latency type but instead adds local latency and network latency together. This provides a total of nine (9) different total latencies: 25, 100, 125, 175, 200, 225, 275, 300, and 375 milliseconds.

5.3 Player Performance

Figure 6 depicts player score percent versus total latency. The axes, data and trendlines are as for Figure 5 but the datasets are not differentiated based on latency type. From the graph, there is generally a downward linear trend in player performance with latency – i.e., players perform worse with higher latency. The linear regression fits the mean values well, with an R^2 of 0.93 and $p < .001$. As a take-away, an increase in total latency by 100 milliseconds decreases score percent by 4.4.

For a comparison with other navigation work, Friston et al. [17] and Pantel and Wolf [37] show degraded user performance for steering tasks, about 30% and 50%, respectively, for 100 milliseconds of latency. Considering navigation as part of first-person shooter gameplay, Liu et al. [30] find player scores degrade about 20% for 100 milliseconds of latency. The impact of latency on player performance is considerably higher for these other games/tasks than it is for Hide and Seek, possibly because they require strict adherence to a fixed path [17, 37] or include a target selection (shooting) component in addition to navigation. In contrast, Hide and Seek does not have paths to follow nor targets to select – movement can be anywhere in the room and any "selection" is done by just looking at the opponent.

5.4 Quality of Experience

Quality of Experience (QoE) was assessed from the user responses to a Mean opinion Score (MOS) question filled out at the end of each round. Responses are converted to a 5 point scale, from 1-low to 5-high.

Figure 7 depicts QoE versus latency. The x-axes and trendline are as for Figure 6, but the data here are the QoE responses, shown on the y-axis, instead of the score. From the graph, mean user QoE degrades with latency. The linear regression fits the means well, with R^2 0.95 and $p < .001$. As a take-away, an increase in total latency by 100 milliseconds decreases player QoE by 0.5 points on a 5-point scale.

While linear trends fit both Figure 6 and Figure 7 well, we note there are sub-regions where the linear trend does not clearly hold. For 100 - 175 milliseconds and 200 - 275 milliseconds of total latency, player performance does not vary much with latency, nor does QoE. Future work could analyze if and how sub-ranges of latency deviate from the overall linear trend.

5.5 Hider and Seeker Times

In first-person shooter games, what often matters is how long a player can see or be seen by an opponent. Longer time windows make it more likely to get a target in sight, aim, shoot and hit. For

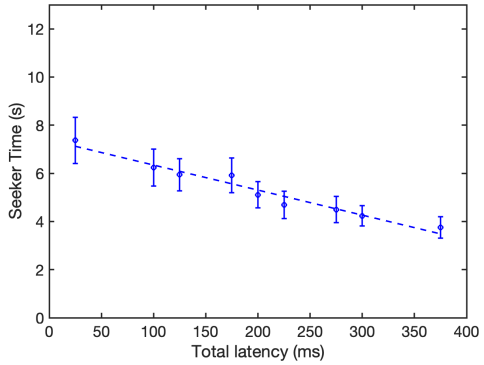

Figure 8: Seeker time versus latency.

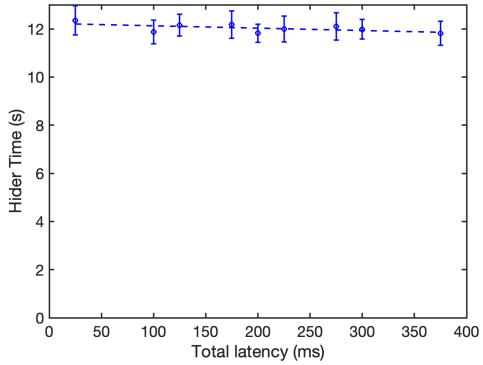

Figure 9: Hider time versus latency.

our game, seeker time is the total round time that the player has the opponent in sight while seeking, and hider time is the total round time that the player is out of the opponent's sight while hiding. We convert hider and seeker values to "per minute" rather than per round (i.e., dividing times by 40 seconds) for ease of understanding.

Based on our user study results, hiding is "easier" than seeking, as a hider spends more time being hidden than the seeker spends seeing the opponent. With 25 milliseconds of latency, a hider is hidden about 62% of the time, while a seeker sees the opponent about 38% of the time.

Figure 8 depicts seeker time versus latency and Figure 9 depicts hider time versus latency. The x-axes and trendlines are as for Figure 6, but the data and y-axes here are the seeker time in a round (Figure 8) and hider time in a round (Figure 9). From Figure 8, the seeker time gets shorter with latency, meaning the player sees less of the opponent with an increase in latency. The linear regression fits the means well, with R^2 0.96 and $p < .001$. As a take-away, an increase in total latency by 100 milliseconds degrades seeker time by 1.5 seconds per minute. Contrast this to the hider time in Figure 9. The hider time is relatively constant with latency, as indicated by the mostly flat regression line. The linear regression has an R^2 of only 0.34 and $p = 0.1$. This indicates that the ability of a player to hide from an opponent is not significantly impacted by latency.

Put together, latency would appear to have the strongest effect on navigation when a player is maneuvering to see an opponent but a much more limited effect on navigation when a player is avoiding being seen. As a possible explanation, for both roles, generally the best strategy is to stay near an obstacle since this lets a player peek/duck around the corner with relatively small movements. However, when an opponent is hiding around a distant pillar, the seeking player must move much further to get into position to see them, which, in turn, requires more movement (WASD) keypress actions. Assuming latency degrades performance for each keypress action, the cumulative effect of latency on the keypress actions means a greater decrease in seeker time than for hider time when there is latency. This finding also aligns to the preference of latency compensation techniques for first-person shooter games that favor giving the correct behavior for a shooter (akin to a seeker in our game) versus the target (akin to a hider in our game) [6]. Note, this

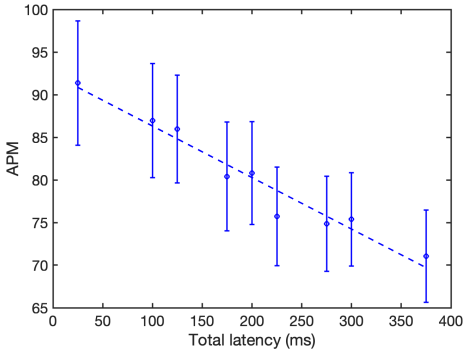

Figure 10: Actions per minute versus latency.

does *not* mean that seekers move less than hiders (in fact, both roles have the same action rate – see Section 5.6), but rather that more of the hider's actions are taken while already hidden in comparison to the seeker's actions which mostly occur while trying to spot the opponent.

5.6 Actions per Minute

Actions per minute has been proposed as one metric to classify a game's sensitivity to latency [43]. We analyze the converse – whether latency affects the player's actions per minute. For navigation, the core parameter is how often the player intentionally moves in an intended direction, and for Hide and Seek (and most first-person navigation games on a PC) this is via the WASD keys. We compute actions per minute by the number of times a player presses WASD keys in a round divided by the round length (40 seconds) multiplied by 60.

Figure 10 depicts actions per minute versus latency. The x-axes and trendlines are as for Figure 6, but the data and y-axes here are the actions per minute. In general, there is considerable variation in actions per minute each round (as can be seen by the sizes of the confidence intervals), more so than for score (Figure 6) and considerably more so than for QoE (Figure 7). There is also a noticeable downward trend as latency increases. The linear regression fits the means well with $R^2 = 0.95$ and $p < .001$. This indicates players

move less often with an increase in latency, possibly because the lower responsiveness requires more deliberate movement actions by the players since players must wait longer for the results of a previous action before applying the next one.

5.7 Modeling

Analytic models can help generalize results beyond the necessarily narrow range of conditions tested in a user study. In our case, this means generalizing to latencies that are not one of the 9 discrete values used in our experiments. Moreover, a model of navigation with latency may be combined with models of target selection with latency [26] in order to simulate moving and shooting in a first-person shooter game. Once validated, this has the potential to enable predictions of player performance over a broad range of first-person shooter game configurations.

Since the intent of navigation during first-person shooter combat is to get into position to shoot (or avoid being shot), we analyze and then model the individual hider and seeker time intervals (i.e., duration that the other player is visible) with latency. The former represents time windows when the player is hidden and cannot be shot, while the latter represents time windows when the player can see the opponent and potentially shoot them. In both cases, longer is better – more time being hidden or more time seeing a target. The cumulative distribution functions (CDFs) of seeker and hider intervals appear to be exponential, but with a heavy tail, thus we fit a Weibull distribution to the data. The CDF of a Weibull distribution is:

$$1 - e^{-(x/\lambda)^k} \tag{1}$$

where k is the shape parameter and λ is the scale parameter. Details on the fit models are shown in Table 4. For all intervals, seeker and hider, a Weibull fits well with R^2 0.99 and RMSE of 0.01. Autocorrelation results suggest that the length of an interval does not correspond to the length of intervals that follow.

Future work should validate and ascertain if the model results hold for other conditions not tested. In particular, the results may be quite dependent upon the map configuration (e.g., number of obstacles) as well as the speeds and the sizes of the avatars.

Table 4: Modeling results.

interval distribution	λ	k	R^2	RMSE
All	0.29	0.61	0.99	0.01
Seeker	$0.0005 \cdot l + 0.27$	$0.0003 \cdot l + 0.63$	0.99	0.01
Hider	0.50	0.72	0.99	0.02

6 LIMITATIONS AND FUTURE WORK

As noted in Section 4, our user study had 30 users in total. While this sample size was large enough for statistically significant results for user performance and quality of experience with latency, more users would tighten the confidence bounds in Figure 6 and Figure 7. Similarly, potentially sampling more latencies, especially within the ranges we currently study, could help determine where linear trends do and do not hold.

Our sample is skewed towards males (only 4 females out of 30 participants). While this may reflect the gender breakdown present in some first-person shooter games today, the results reported may not be representative of female performance. A follow-on study might also screen users for expertise in first-person games (e.g., using self-rated skill [27]) in order to provide for more focused analysis.

Our methodology intentionally had users compete against the same opponent in order to provide consistency across game conditions, save for the latency. This means, however, that our results are based on a specific opponent skill level – the impact of latency combined with different opponent skills was not assessed. Our use of the same human opponent for all users may limit the reproducibility of the study. While we would expect that trends would hold for other human opponents, the relative amounts may differ.

Commercial first-person shooter games often incorporate latency compensation techniques [31] to mitigate the effects of network latency on players. Thus, our results may not generalize to commercial first-person games with network latency. However, since many if not most latency compensation techniques cannot be used to overcome local latencies nor network latencies in cloud-based game streaming systems, our results should still be relevant for navigation in many game configurations. Plus, many non-commercial games with navigation do not implement latency compensation techniques due to their (the technique's) complexity – our results are relevant for these, too.

Serious game players often customize the software settings on their computers and games to suit their personal play preferences. For example, players may alter the mouse sensitivity or change the graphics resolution from the system defaults. These custom changes presumably improve that player's experience and/or performance. However, since customizations that deviated from our settings create a difference in test conditions between users, we did not allow any changes to the computer settings. This holds for other game configurations, too, such as using other mice, keyboards or monitors.

As noted in Section 5.7, future work is to validate the hiding and seeking models, then combine those with target selection models to simulate first-person shooter player performance. Once the simulations are validated, they could be used to assess first-person shooter player performance over a wide range of system and game configurations.

Other future work could apply the same methodology used in our paper to player actions in other games genres, e.g., Multiplayer Online Battle Arena (MOBA) games like *DOTA 2* (Valve, 2013) and *League of Legends* (Riot Games, 2019), and Real-Time Strategy (RTS) games like *Starcraft* (Blizzard, 1998). For this, individual game actions would need to be isolated for the game and then evaluated, such as navigation for moving an avatar from a third-person perspective.

7 CONCLUSION

People are increasingly turning to games for entertainment evidenced by the growth in the game and esports industries, particularly so during the COVID-19 pandemic. Many multi-player computer games connect players via a network, meaning network

latency delays player actions from providing input to viewing the corresponding output, and this delay comes in addition to any local latency from the game system. While previous studies have assessed the impact of latency on computer games, in general, and first-person shooter games specifically, the degree to which latency impacts individual actions within a game is not well known. In particular, while latency's effect on target selection has been studied and can be used as a proxy for aiming in a first-person shooter game, latency's effect on navigation – moving an avatar in a virtual world – is relatively unknown. Understanding the effects of latency on first-person navigation can help inform game design and development techniques to mitigate latency's effects, and also has the potential to generalize results to a broad range of first-person games through modeling, validation and simulation.

Analysis of results from our first-person navigation user study shows that there is no significant difference in the impact of local latency versus network latency on player navigation performance in the absence of latency compensation techniques. Across the range of total latencies studied, player performance and quality of experience (QoE) both degrade linearly as latencies increase from 25 milliseconds to 375 milliseconds. Specifically, player scores at 25 milliseconds average over 25% better than player scores at 375 milliseconds. Over this same range, QoE decreases even more (nearly 60%), with the QoE at 25 milliseconds being about 4 (on a 5 point scale) and the QoE at 375 milliseconds falling to about 2.2. Player ability to move into position to see opponents decreases with latency; however player ability to hide to avoid being seen does not vary much with latency. The rate of player game actions decreases with latency by about 30% from 25 milliseconds to 375 milliseconds of latency.

REFERENCES

[1] Johnny Accot and Shumin Zhai. 1997. Beyond Fitts' Law: Models for Trajectory-Based HCI Tasks. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '97)*. Association for Computing Machinery, New York, NY, USA, 295–302. https://doi.org/10.1145/258549.258760

[2] Johnny Accot and Shumin Zhai. 1999. Performance Evaluation of Input Devices in Trajectory-Based Tasks: An Application of the Steering Law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. Association for Computing Machinery, Pittsburgh, Pennsylvania, USA, 466–472. https://doi.org/10.1145/302979.303133

[3] 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*. Springer-Verlag, Berlin, Heidelberg, 97–106.

[4] Grenville Armitage. 2003. An Experimental Estimation of Latency Sensitivity in Multiplayer Quake 3. In *Proceedings of the 11th IEEE International Conference on Networks (ICON)*. IEEE, Sydney, Australia, 137–141. https://doi.org/10.1109/ICON.2003.1266180

[5] Tom Beigbeder, Rory Coughlan, Corey Lusher, John Plunkett, Emmanuel Agu, and Mark Claypool. 2004. The Effects of Loss and Latency on User Performance in Unreal Tournament 2003®. In *Proceedings of 3rd ACM SIGCOMM Workshop on Network and System Support for Games* (Portland, Oregon, USA) *(NetGames '04)*. Association for Computing Machinery, New York, NY, USA, 144–151. https://doi.org/10.1145/1016540.1016556

[6] Yahn W. Bernier. 2001. Latency Compensating Methods in Client/Server In-game Protocol Design and Optimization. In *Proceedings of the Game Developers Conference (GDC)*. CMP Media, LLC, San Francisco, CA, USA, 13 pages.

[7] Costas Boletsis, Jarl Erik Cedergren, and Marco Porta. 2019. VR Locomotion in the New Era of Virtual Reality: An Empirical Comparison of Prevalent Techniques. *Advances in Human-Computer Interaction* 2019 (jan 2019), 15 pages. https://doi.org/10.1155/2019/7420781

[8] Tim Buker, Dennis Vincenzi, and John Deaton. 2012. The Effect of Apparent Latency on Simulator Sickness While Using a See-Through Helmet-Mounted Display: Reducing Apparent Latency With Predictive Compensation. *Human Factors* 54 (04 2012), 235–49. https://doi.org/10.1177/0018720811428734

[9] Jean-Remy Chardonnet, mohammad Mirzaei, and Frédéric Merienne. 2021. Influence of Navigation Parameters on Cybersickness in Virtual Reality. *Virtual Reality* 25, 3 (Sept. 2021), 565–574. https://doi.org/10.1007/s10055-020-00474-2

[10] Kajal Claypool and Mark Claypool. 2007. On Frame Rate and Player Performance in First Person Shooter Games. *Springer Multimedia Systems Journal (MMSJ)* 13 (2007), 3–17. DOI 10.1007/s00530-007-0081-1.

[11] Mark Claypool. 2018. Game Input with Delay—Moving Target Selection with a Game Controller Thumbstick. *ACM Trans. Multimedia Comput. Commun. Appl.* 14, 3s, Article 57 (June 2018), 22 pages. https://doi.org/10.1145/3187288

[12] Mark Claypool, Andy Cockburn, and Carl Gutwin. 2020. The Impact of Motion and Delay on Selecting Game Targets with a Mouse. *ACM Trans. Multimedia Comput. Commun. Appl.* 16, 2s, Article 73 (June 2020), 24 pages. https://doi.org/10.1145/3390464

[13] Mark Claypool, Ragnhild Eg, and Kjetil Raaen. 2017. Modeling User Performance for Moving Target Selection with a Delayed Mouse. In *Proceedings of the 23rd International Conference on MultiMedia Modeling (MMM)*. Springer International Publishing, Reykjavik, Iceland, 226–237.

[14] Matthias Dick, Oliver Wellnitz, and Lars Wolf. 2005. Analysis of Factors Affecting Players' Performance and Perception in Multiplayer Games. In *Proceedings of 4th ACM Workshop on Network and Systems Support for Games (NetGames)*. Association for Computing Machinery, Hawthorn, NY, USA, 1–7.

[15] E$ports Earning. 2019. Prize Money, Results, History, Statistics. Online: https://www.esportsearnings.com/. (Accessed January 5, 2021).

[16] 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.

[17] 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 (2016), 1605–1615. https://doi.org/10.1109/TVCG.2015.2446467

[18] Tobias Fritsch, Hartmut Ritter, and Jochen H. Schiller. 2005. The Effect of Latency and Network Limitations on MMORPGs: a Field Study of Everquest 2. In *Proceedings of the 4th ACM Network and System Support for Games (NetGames)*. Association for Computing Machinery, Hawthorne, NY, USA, 1–9.

[19] Christina Gough. 2020. eSports Audience Size Worldwide from 2018 to 2023. Statista. Online: https://tinyurl.com/y3tffxzo. (Accessed September 17, 2020).

[20] Oliver Hohlfeld, Hannes Fiedler, Enric Pujol, and Dennis Guse. 2016. Insensitivity to Network Delay: Minecraft Gaming Experience of Casual Gamers. In *Proceedings of the International Teletraffic Congress (ITC)*. IEEE, Würzburg, Germany, 31–33.

[21] 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 Conference on Human Factors in Computing Systems (CHI)*. Association for Computing Machinery, Seoul, Republic of Korea, 135–144.

[22] Raghavendra S. Kattinakere, Tovi Grossman, and Sriram Subramanian. 2007. *Modeling Steering within Above-the-Surface Interaction Layers*. Association for Computing Machinery, New York, NY, USA, 317–326. https://doi.org/10.1145/1240624.1240678

[23] Nick Lee. 2017. Beyond 50/50: Breaking Down The Percentage of Female Gamers by Genre. Online: https://quanticfoundry.com/2017/01/19/female-gamers-by-genre/. (Accessed September 5, 2021).

[24] Wai-Kiu Lee and Rocky K. C. Chang. 2015. Evaluation of Lag-Related Configurations in First-Person Shooter Games. In *Proceedings of ACM/IEEE Workshop on Network and Systems Support for Games (NetGames)*. IEEE Press, Zagreb, Croatia, Article 11, 3 pages.

[25] Shengmei Liu and Mark Claypool. 2021. EvLag - A Tool for Monitoring and Lagging Linux Input Devices. (2021). (in press).

[26] Shengmei Liu and Mark Claypool. 2021. *Game Input with Delay – A Model of the Time Distribution for Selecting a Moving Target with a Mouse*. Springer International Publishing, Prague, Czech Republic, 506–518. https://doi.org/10.1007/978-3-030-67832-6_41

[27] Shengmei Liu, Mark Claypool, Bhuvana Devigere, Atsuo Kuwahara, and Jamie Sherman. 2020. 'Git Gud!' – Evaluation of Self-Rated Player Skill Compared to Actual Player Performance. In *Extended Abstracts of the 2020 Annual Symposium on Computer-Human Interaction in Play* (Virtual Event, Canada) *(CHI PLAY '20)*. Association for Computing Machinery, New York, NY, USA, 306–310. https://doi.org/10.1145/3383668.3419906

[28] Shengmei Liu, Mark Claypool, Atsuo Kuwahara, James Scovell, and Jamie Sherman. 2021. The Effects of Network Latency on Competitive First-Person Shooter Game Players. In *2021 13th International Conference on Quality of Multimedia Experience (QoMEX)*. IEEE press, Virtual, 151–156. https://doi.org/10.1109/QoMEX51781.2021.9465419

[29] Shengmei Liu, Atsuo Kuwahara, James Scovell, Jamie Sherman, and Mark Claypool. 2021. Comparing the Effects of Network Latency versus Local Latency on Competitive First Person Shooter Game Players. https://doi.org/10.31219/osf.io/bhxcy

[30] Shengmei Liu, Atsuo Kuwahara, James Scovell, Jamie Sherman, and Mark Claypool. 2021. Lower is Better? The Effects of Local Latencies on Competitive First-Person Shooter Game Players. In *Proceedings of ACM CHI*. ACM SIGCHI, Yokohama, Japan, 12 pages.

[31] Shengmei Liu, Xiaokun Xu, and Mark Claypool. 2022. A Survey and Taxonomy of Latency Compensation Techniques for Network Computer Games. *Comput. Surveys* 1, 1 (Feb. 2022), 37 pages. https://doi.org/10.1145/3519023

[32] Michael Long and Carl Gutwin. 2018. Characterizing and Modeling the Effects of Local Latency on Game Performance and Experience. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play (CHI Play)*. Association for Computing Machinery, New York, NY, USA, 285–297.

[33] Michael Long and Carl Gutwin. 2019. Effects of Local Latency on Game Pointing Devices and Game Pointing Tasks. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI)*. Association for Computing Machinery, Glasgow Scotland, UK, 1–12.

[34] 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, The Netherlands, Amsterdam, Netherlands, 6 pages.

[35] Emmie Martin. 2018. Super Bowl Champs Will Win Thousands - but They'd Earn 130 Percent More If They Played Baseball. CNBC - Money Online: https://www.cnbc.com/2018/02/02/how-much-super-bowl-winners-get-paid-compared-to-world-series.html. (Accessed January 5, 2021).

[36] optimum.com. 2020. What Is Latency? Online: https://www.optimum.com/internet/what-latency. (Accessed Nov 15, 2021).

[37] Lothar Pantel and Lars C. Wolf. 2002. On the Impact of Delay on Real-Time Multiplayer Games. In *Proceedings of the Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*. Association for Computing Machinery, Miami, FL, USA, 23–29.

[38] Andriy Pavlovych and Carl Gutwin. 2012. Assessing Target Acquisition and Tracking Performance for Complex Moving Targets in the Presence of Latency and Jitter. In *Proceedings of Graphics Interface 2012* (Toronto, Ontario, Canada) *(GI '12)*. Canadian Information Processing Society, CAN, 109–116.

[39] Andriy Pavlovych and Wolfgang Stuerzlinger. 2011. Target Following Performance in the Presence of Latency, Jitter, and Signal Dropouts. In *Proceedings of Graphics Interface 2011* (St. John's, Newfoundland, Canada) *(GI '11)*. Canadian Human-Computer Communications Society, Waterloo, CAN, 33–40.

[40] Peter Quax, Patrick Monsieurs, Wim Lamotte, Danny De Vleeschauwer, and Natalie Degrande. 2004. Objective and Subjective Evaluation of the Influence of Small Amounts of Delay and Jitter on a Recent First Person Shooter Game. In *Proceedings of 3rd ACM Workshop on Network and Systems Support for Games (NetGames)*. Association for Computing Machinery, Portland, OG, USA, 152–156.

[41] Kjetil Raaen and Ragnhild Eg. 2015. Instantaneous Human-Computer Interactions: Button Causes and Screen Effects. In *Proceedings of the 17th HCI International Conference*. Springer International Publishing, Los Angeles, CA, USA, 492–502.

[42] Saeed Shafiee Sabet, Steven Schmidt, Saman Zadtootaghaj, Carsten Griwodz, and Sebastian Moller. 2018. Towards Applying Game Adaptation to Decrease the Impact of Delay on Quality of Experience. In *IEEE International Symposium on Multimedia (ISM)*. IEEE press, Taichung, Taiwan, 114–121.

[43] Saeed Shafiee Sabet, Steven Schmidt, Saman Zadtootaghaj, Carsten Griwodz, and Sebastian Möller. 2020. Delay Sensitivity Classification of Cloud Gaming Content. In *Proceedings of the 12th ACM International Workshop on Immersive Mixed and Virtual Environment Systems*. Association for Computing Machinery, New York, NY, USA, 25–30. https://doi.org/10.1145/3386293.3397116

[44] Stephen Serge and Jason Moss. 2015. Simulator Sickness and the Oculus Rift: A First Look. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 59 (09 2015), 761–765. https://doi.org/10.1177/1541931215591236

[45] Robert J. Teather, Andriy Pavlovych, Wolfgang Stuerzlinger, and I. Scott MacKenzie. 2009. Effects of Tracking Technology, Latency, and Spatial Jitter on Object Movement. In *Proceedings of the IEEE Symposium on 3D User Interfaces*. IEEE press, Lafayette, LA, USA, 43–50.

[46] wepc.com. 2021. Video Game Industry Statistics, Trends and Data In 2021. Online: https://www.wepc.com/news/video-game-statistics/. (Accessed August 12, 2021).

[47] Shumin Zhai, Johnny Accot, and Rogier Woltjer. 2004. Human Action Laws in Electronic Virtual Worlds: An Empirical Study of Path Steering Performance in VR. *Presence* 13 (04 2004), 113–127. https://doi.org/10.1162/1054746041382393