WPI-CS-TR-WPI-CS-TR-22-02 Apr 2022

Waiting to Play – Measuring Game Load Times and their Effects on Players

by

Shengmei Liu
Federico Galbiati
Miles Gregg
Eren Eroglu
Atsuo Kuwahara
James Scovell
Mark Claypool

Computer Science
Technical Report
Series

WORCESTER POLYTECHNIC INSTITUTE

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

Waiting to Play – Measuring Game Load Times and their Effects on Players

Shengmei Liu, Federico Galbiati, Miles Gregg, Eren Eroglu, Mark Claypool
sliu7,fgalbiati,mgregg,ezeroglu,claypool@wpi.edu
Worcester Polytechnic Institute, Worcester, MA
USA

Atsuo Kuwahara, James Scovell
atsuo.kuwahara,james.j.scovell@intel.com
Intel Corporation, Hillsboro, OR
USA

ABSTRACT

Before playing, gamers must wait for the game to launch the level to load, waiting for at least several seconds for some games but up to several minutes for others. While the effects of waiting on user experience is well-studied for some domains, such as Web browsing, the effects of wait times on games players is not known, nor is the impact of computer system components, such as the processor or graphics card, on game loading times. We present results from two experiments: 1) a user study that evaluates the impact of game loading time using a custom tool that simulates game loading and collects player experience, and 2) a systematic measurement of the effects of key system components on the game load times themselves. Analysis of the user study results shows game loading time has a pronounced effect on player quality of experience, but differs based on game time and game load content. Analysis of the measurement experiments shows the potential to reduce game load times – processor and graphics card have a significant effect on game load times, but storage device less so.

CCS CONCEPTS

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

KEYWORDS

gamer, games, user study, QoE, hardware, measurement

ACM Reference Format:
Shengmei Liu, Federico Galbiati, Miles Gregg, Eren Eroglu, Mark Claypool and Atsuo Kuwahara, James Scovell. 2022. Waiting to Play – Measuring Game Load Times and their Effects on Players. In *ACM Multimedia, October 10-14, 2022, Lisbon, Portugal.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1234/1234567.1234567

1 INTRODUCTION

Human-computer interactions often have moments where users must wait for the system to gather the application data before users can interact with it. Common examples include waiting for a file to download, a Web page to load or a video to finish buffering before the user experience can commence. Early studies in human-computer interaction converged on a commonly agreed "acceptable" waiting threshold of about ten seconds, beyond which users find waiting reduces the quality of the experience [2, 9]. Longer waiting times can be sources of boredom, at best, and anxiety and frustration at worst. While prior research has demonstrated the relationship between user waiting time and decreased user satisfaction [3, 7, 8], the severity of this degredation depends upon the application [6].

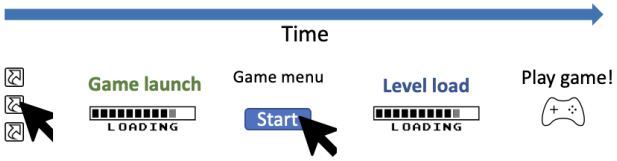

Figure 1: The player's perspective of game loading.

Computer game players often face considerable wait times since the game needs to load before play can commence. During game loading, the computer system reads and converts data needed by the game engine and initializes various sub-systems, such as physics and graphics, needed to run the game. For the player, loading a game happens during two phases, depicted in Figure 1. The player must first wait while the game is launched (labeled "Game Launch"), much as is done for any application started on a computer. Once the game has been launched, the player enters a home screen which has an interface to select the version of the game to play (e.g., game mode and character selection). Once the player has configured all game settings, the player must again wait while the game level loads (labeled "Level Load"). Only once the level had loaded can play commence.

To mitigate the negative effects of waiting on users, applications can provide feedback on progress, widely accepted as a means to improve a user's waiting experience [9, 11]. Feedback during game loading can take many forms such as progress bars, text messages, and animated graphics and videos.

In addition, while decreasing the waiting time for some Internet-based applications can be done by increasing network bitrates [5], much of what a computer game loads is not related to the network capacity. It is unknown how standard hardware upgrades, such as improved processor, storage device or graphics card, can decrease game load times.

This paper presents seeks to better understand game load times and quality of experience and game load times and hardware components via two experiments. The first is a user study that assesses

ACM MM '22, October 10-14, 2022, Lisbon, Portugal

ACM ISBN 978-1-XXXX-XXXX-1/1/22…$15.00
https://doi.org/10.1234/1234567.1234567

the quality of experience for users waiting for a game to load. We selected 12 popular games and encoded their game loadings as videos, re-encoding them to different lengths. We then recruited 54 users that had experience playing at least 3 of the games and had each participant use our custom application to simulate loading each game and provide an assessment of the experience for each. Analysis of the results shows a marked decrease in quality of experience (QoE) with game load times with results that vary from game to game. Models based on game loading time, possibly supplemented with visual content, have the potential to accurately predict QoE for loading games in general without needing to measure the QoE of individual games.

Based on these results, players and computer system developers may see benefits to user quality of experience from decreasing game load times. So, the second experiment measures game load times on different computer systems, varying processor, graphics card and storage device for the 12 games tested in the user study. Analysis of the measurement experiments shows game load times vary considerably (almost a magnitude) across games, and processor and graphics device have a significant effect on game load times, while storage device type less so.

The rest of the report is organized as follows: Section 2 describes what happens in the computer system during game loading; Section 3 provides work related to this paper; Sections 4 and 5 detail our methods and results for our user study and measurements experiments, respectively; Section 6 mentions some limitations of our work; and Section 7 summarizes our conclusions.

2 GAME LOADING BACKGROUND

During game loading, the game system must transfer and convert game data from long-term storage, such as a hard drive or a solid state drive, to memory so it can be used by the game software. The largest volume of game data is typically from art assets, such as models, animations, and textures, but also includes sound effects and music. These assets usually need to be converted from their storage format, typically compressed to save space, to an in-memory format usable by the game engine. Other aspects of game loading include bringing in and parsing game data such as virtual world maps and game object attributes, and code and data for game systems and interfaces to control them. Processing during loading also includes pre-calculations and pre-rendering for visuals, and initialization of the many different sub-systems that run the game, such as the graphics, physics and AI engines, as well as networking and logging. Garbage collection may also run in order to free up code and assets that are used during initialization but can subsequently be discarded.

For the player, the above process of loading a game happens during two phases, depicted in Figure 1. The player must first wait while the game is launched (labeled "Game Launch"), much as is done for any application started on a computer. Once the game has been launched, the player enters a home screen which has an interface to select the version of the game to play (e.g., game mode and character selection). Once the player has configured all game settings, the player must again wait while the game level loads (labeled "Level Load"). Only once the level had loaded can play commence.

So, the player must wait during two phases of game loading: game launch and level load. Game launch happens only once, when the game is first started and does not need to happen again until the player exits the application and restarts at a later time. Level load, however, needs to happen each time the player starts or re-starts a game level which can be as frequently as once every few minutes for some games or as infrequently as once every few hours, for others.

During both phases of game loading – game launch and level load – the game often shows visuals, such as splash screens with company logos from the game studios, publishers or systems, or images of game characters or game scenes, or even cinematic renditions of gameplay, sometimes accompanied by sound effects or music. In some cases, the cinematics and animations can be skipped by the player with the press of a button or key. Some load screens show a progress indicator, such as a loading bar or animated loading icon, providing feedback to the player that the game system is getting the game ready to play.

3 RELATED WORK

This section describes related work in three areas: user experience while waiting 3.1, feedback to mitigate waiting 3.2 and cognitive load and waiting 3.3.

3.1 Wait Time and User Experience

Egger et al. [5] discuss the psycho-physics basis for a model of quality of experience and human time perception (e.g., delay when buffering a video during initial playout). The authors describe a set of studies that lay out the basis for a logarithmic relationship for waiting time and user satisfaction ratings. They find this relationship holds for several tasks, including file downloading – i.e., that user satisfaction with file downloads decreases logarithmicly with download time.

Hoßfeld et al. [6] quantify the impact of wait time on user experience for different application scenarios by means of subjective laboratory and crowdsourcing studies. They find user quality of experience (QoE) for the waiting time depends on the application, with users waiting for a video to start playing being more tolerant of waiting than users logging into a social network.

Allard et al. [1] study the tradeoffs between initially waiting for a streaming video to start playing and interrupts (waiting) in the middle of a video playout. They show users' annoyance with waiting increases logarithmically with the time it takes a video to start playing, with this annoyance greater for interrupts.

3.2 Feedback While Waiting

Branaghan and Sanchez [3] found that duration indicators for users watching simulated movies were most satisfactorily indicated with a constant progress bar. Nah [8] examined how feedback while waiting improves user satisfaction and showed the presence of a display feedback increases how long a user is willing to wait. Lallemand and Gronier [7] confirm these results and show users more satisfied with a waiting interface that provides more information. However, too much feedback with details on progress can make a wait seem longer as the user interprets every event as taking time, and more events feels like more waiting time [3, 7].

3.3 Cognitive Load While Waiting

Lalleman and Gronier [7] use cognitive models of time perception, varying the cognitive workload and informational feedback, to study impact on satisfaction and perceived waiting time for users. They find a link between cognitive workload and waiting time perception. Users judged shorter waiting times more positively with a decrease in satisfaction that is linear with time.

4 QUALITY OF EXPERIENCE

In order to assess the effects of game load time on player experience, we selected games to cover a range of genres, designed and implemented an application to simulate game loading, recruited participants for a user study, had each participant load games and rate their quality of experience, and analyzed the results.

4.1 Methodology

Table 1: Games studied

Game	Publisher	Year	Genre
ACOD [13]	Ubisoft	2018	First-Person Shooter
Apex Legends [12]	EA	2019	First-Person Shooter
Civ VI [14]	2K	2016	Turn-based Strategy
CS:GO [15]	Valve	2012	First-Person Shooter
Fortnite [16]	Epic	2017	Battle Royale
GTA V [17]	Rockstar	2013	Role-Playing Game
Hearthstone [18]	Blizzard	2014	Turn-based Strategy
LoL [19]	Riot	2009	MOBA
Minecraft [20]	Mojang	2011	Sandbox
Overwatch [21]	Blizzard	2016	First-Person Shooter
PUGB [22]	Microsoft	2017	Battle Royale
R6S [23]	Ubisoft	2015	First-Person Shooter

We selected twelve (12) games based on their popularity to make it more likely participants in our sample pool would have some familiarity, while also covering a range of game types. Table 1 shows the abbreviated name for the games (the full names can be found in the references), listed in alphabetic order along with publisher, year published and genre. The twelve games cover six (6) genres: four first-person shooter games, two turn-based strategy games, two battle royale games, two role-playing games, one multiplayer online battle arena (MOBA) game and one sandbox game. For each game, we record a video of the game launch and the level load on the same PC in order to provide for a single point of reference. The PC is an Alienware with an Intel i7-6700 CPU @3.4 GHz with 16 GB RAM and an NVIDIA GeForce GTX 1070 graphics card.

In order to assess how changes to loading times might impact player experience (e.g., if the user purchased a faster computer system or if the game developer reduced the game load time), we scaled the lengths of the videos for each game: 0.25, 0.5, 0.75, 1.0 and 1.25. So, for example, a 10 second game load video scaled to 0.25 would run in 2.5 seconds. When scaling, we took care to reduce the times only for non-animated portions of the load videos in order to minimally distort the visuals.

In order to mimic the game loading experience, we developed a stand-alone application that had users launch the game (e.g., double-click on the game icon on the desktop) and load the level (e.g., select "start" and pick the intended map to play) as if they were about to play the game. For each case – game launch and level load – the application plays the pre-recorded video, pausing when input is needed by the user to proceed. When this happens, the app highlights where the user must click (e.g., the "start" button). When the video finishes playing, the application pops up a survey to assess the players' quality of experience via two questions: a) a Mean Opinion Score (MOS) "Please rate your experience" with a text box for 1.0 to 5.0 point numeric entry, shown along with scale: Excellent, Good, Fair, Poor, Bad; and b) a binary question "Is the experience acceptable?"

Since a user's opinion of a game load's time may depend upon their familiarity with the game, users were invited to participate in the study based on how much and how recently they played each game in Table 1. Invited users were familiar with at least 3 of the games, chosen so as to provide an approximately equal number of users for all games. After testing, the number of users for each game was:

ACOD (15), Apex Legends (15), Civ VI (18), CS:GO (19), Fortnite (18), GTA V (18), Hearthstone (16), LoL (17), Minecraft (19), Overwatch (17), PUGB (16), and R6S (17).

For each participant, the application randomly shuffled the order of the games loaded. For each game, users first assessed all the game launches and then assessed all the level loads, with the scale lengths randomly shuffled. Thus, we had a between-subjects design, where each user loaded 3 (of the possible 12) games, each with two conditions (game launch and level load) and 5 lengths for each condition ($3 \times 2 \times 5$) for a total of 30 game loads.

The user study was conducted in a dedicated, on-campus computer lab. Our test computer was a Windows 10 Alienware with an Intel i7-4790K CPU @4 GHz with 16 GB RAM and an Intel HD 4600 graphics card. While users may have more powerful PCs to play the game, our game load simulation application is lightweight, needing only processing akin to that needed for a streaming video player. However, in order to provide for fast input and display, the PC was equipped with a gaming mouse and high refresh rate monitor: a 25" Lenovo Legion monitor, 1920x1080 16:9 pixels @240 Hz with AMD FreeSync and a 1 ms response time; and b) a Logitech G502 mouse, 12k DPI with a 1000 Hz polling rate.

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

In summary, the procedure each user followed was:

(1) Submit screener to ensure familiarity with the games.
(2) For invited participants, arrive at the dedicated lab at a scheduled time and sign the consent form.
(3) Adjust the computer chair and monitor so as to be comfortably looking at the center of the screen.
(4) Read the instructions regarding the application and controls.
(5) Launch the game and, when done, fill out the corresponding QoE survey. Repeat for each scaled length (shuffled).

Table 2: Participant demographics

Users	Age (yrs)	Gender	Gaming per Week (hours)	Gamer Self-rating
54	19.3 (1.48)	40 ♂ 13 ♀ 1 ?	10.4 (8.3)	3.4 (1.1)

(6) Load the level and, when done, fill out the corresponding QoE survey. Repeat for each scaled length (shuffled).
(7) Repeat the previous 2 step for each of 3 games (shuffled).
(8) Complete a final demographics questionnaire.

The study length depended upon the games tested, but was under 30 minutes total in nearly all cases. A user study proctor was available for questions and trouble-shooting for the duration.

Study participants were solicited via university email lists. All users were eligible for a $10 USD Amazon gift card upon completion of the study, and many users received playtesting credit for relevant classes in which they were enrolled.

4.2 Analysis

This section first summarizes participant demographics (Section 4.2.1) then presents the core results – Quality of Experience (QoE) (Section 4.2.2) versus game loading time. It then describes derived models of QoE based on game loading time and loading screen content (Section 4.4).

4.2.1 Demographics. Table 2 summarizes the demographic information for the user study participants. Game self-rating is on a five-point scale, 1-low to 5-high. For age and game self-rating, the mean values are given with standard deviations in parentheses. Our user study had 54 participants, ranging from 17-24 years. Gender breakdown is predominantly male (40 males out of 54 users), which reflects the sample pool of students at our university. Half of the participants played 10 or more hours of computer games per week. User self-rating as a gamer ability skews towards above the mid-point (mean 3.4). Most participants majored in Robotics Engineering, Computer Science, or Game Development.

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

Actual game load time. Figure 2 depict scatter plots of QoE ratings versus game load times for game launch and level load. The x-axes are times in seconds and the y-axes are QoE ratings on a 5-point scale (higher is better). Each dot is the QoE value for one user rating one game load. The black dashed lines are linear regression trendlines through all data in each graph. The correlation (R) between rating and time is -0.56 for game launch and -0.58 for level load, indicating a medium strength of downward trend. As a take away, on average, player experience degrades 1.92 points (on a 5-point scale) and 2.58 points for each minute for game launching and level loading, respectively.

Scaled game load time. Since each game load video was scaled relative to the initial encoding, we analyze how QoE changes with relative game load times. Figure 3 depicts the results. The y-axis it

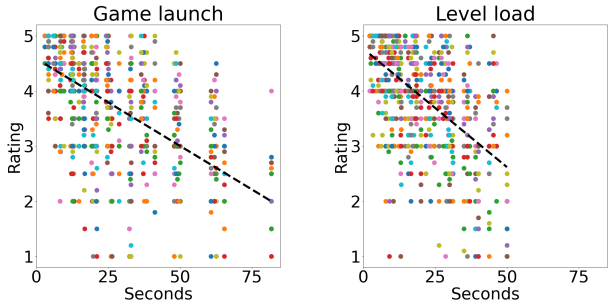

Figure 2: QoE versus game load time (Grouped)

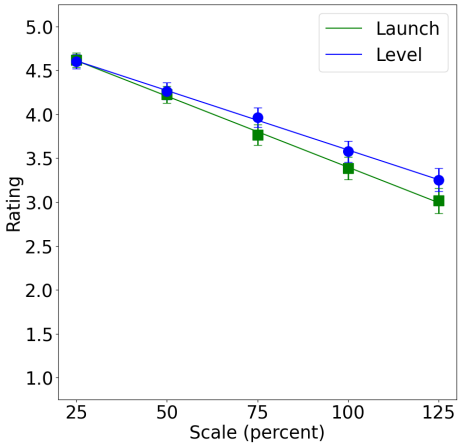

Figure 3: QoE versus game load scale

the QoE rating and the x-axis is the scale relative to the baseline game load time – i.e., 100% is the original scaling, with values below this being faster and values above this being slower. For each loading condition, there are 5 scales ranging from 25% to 125%. The green squares (game launch) and blue circles (level load) are mean values for all users across all games, shown with 95% confidence intervals. The lines are linear regression fits through the mean values. The linear regressions fit the data well for both game launch and level load with R^2 near 1. As a take away, a 50% decrease in game loading time improves player experience by about 0.75 on a 5-point scale, with a slightly larger impact on game launch.

4.3 Threshold and QoE

After each game load, in addition to a QoE rating, players were asked if the experience was acceptable.

Figure 4 depicts the relationship between acceptable and QoE. The x-axis is the QoE rating collected by the MOS question, and the y-axis is the fraction that unacceptable was answered (i.e., "no"). Each dot is the unacceptable fraction for all corresponding QoE values grouped by 0.5 point bins. The green squares are for game

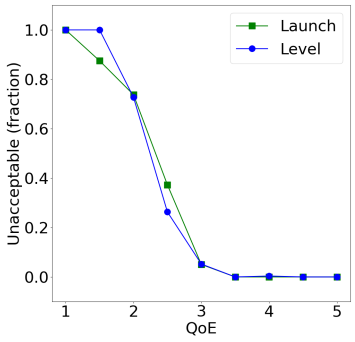

Figure 4: Unacceptability and quality of experience

launch and the blue circles are for level load. In general, QoE values less than 2 are nearly always unacceptable and QoE values above 3.5 are always acceptable and in-between there is a steep change from unacceptable to acceptable. The "above 3.5" threshold can be used in conjunction with QoE models in order to estimate how system improvements that result in reduced game load times pertinent to the user.

4.4 Modeling

This section presents different approaches to modeling QoE for game loading. Such models can be helpful for game and system developers to predict how improvements to game loading time might benefit player experience. Equations for all models presented are not provided inline due to space constraints, but can be found online.[1]

4.4.1 Individual models. While the relationship between QoE and game load time can be coarsely modeled with the "grouped" regression depicted in Figure 2, the individual games themselves may not adhere closely to this relationship.

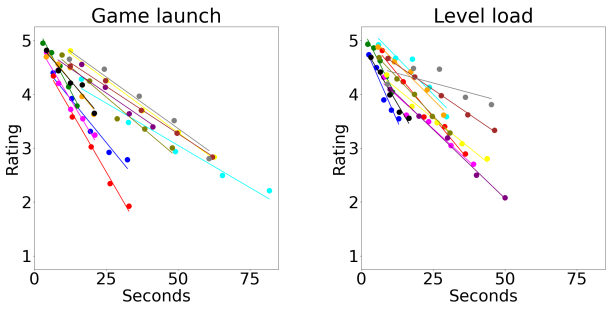

Figure 5: QoE model - Individual

We model QoE for game launch and level load individually for each game. Figure 5 depicts individual models for each game, with axes as in Figure 3. The circles are mean values for all users across the 5 different time scales, shown with 95% confidence intervals. The lines are linear regressions through the mean values. Each color represents data from one game. The regressions fit the individual games well, with R^2 from 0.91 to 1.00 (mean 0.96, SD 0.03) for game launch and R^2 from 0.90 to 1.00 (mean 0.93, SD 0.15) for level load. The exception is for the Apex Legend level load with an R^2 of 0.46.

These individual models represent a sort of "best case" in that the model may accurately reflect QoE for that specific game, but may not generalize to other games.

4.4.2 Unified models. In order to generalize QoE models to games that have not been tested (and may not have even been invented yet), we look to generalize beyond the individual game-specific models.

We use the base load time as a model parameter to derive a "unified model" that can be applied to all games. The unified model fit for game launch is R^2 0.70 and for level load is R^2 0.67, an improvement over the "grouped" model but less than the individual game models.

Since the differences in QoE across games does not appear to be only based on game load times, we use game content as parameters in the model. The ITU recommended [10] measures of spatial perceptual information (SI) and temporal perceptual information (TI) are used as model inputs – for each video, we compute SI and TI for each frame then average. The resulting equations have R^2 0.81 for game launch and 0.82 for level load. Using these model on an untested game would require first capturing the game load content and running it through a tool[2] to compute the SI and TI.

Some game load videos show just one or two screens for the entire load, whereas others have several different scenes to provide visual interest for the user. Similarly, some game loading screens show progress indicators (e.g., a loading bar) that provides feedback to the user about when the game loading might complete. We manually count scene changes and encode progress indicators (0 means none and 1 means one or more) for each game load video and use that as input to the model. Scene changes provide only modest benefit (R^2 0.73 for game launch and R^2 0.67 for level load) as do progress bar indicators (R^2 0.71 for game launch and R^2 0.71 for level load) but together they provide more improvement (R^2 0.75 for game launch and R^2 0.79 for level load). Using these models on an untested game requires first manually watching and scoring the video for scene changes and progress indicators.

Finally, all of SI, TI, scene changes and progress indicators can be used along with the unified model with the base encoding time and actual encoding time. That provides for R^2 0.84 for game launch and R^2 0.85 for level load, within 10% of the individual game load models.

Table 3 summarizes the models ordered by their increasing R^2 values. The clustered and individual models are averaged across the models, with the standard deviation shown in parentheses. The grouped model uses a single equation for QoE based on encoding time for all games. As such, it is the easiest to use, but the least accurate. At the other end are the individual models, one per game, which are quite accurate but require user studies to assess quality with scaling making them impractical. In between are generalized models that unify (U) the base encoding time and actual encoding

[1] (URL hidden to preserve anonymity. Upon acceptance, they will be made public.)

[2] e.g., https://github.com/Telecommunication-Telemedia-Assessment/SITI

time, with options to add scene changes (S) and progress encoding (P) – both of which requires manual scoring – and spatial information (SI) and temporal Information (TI) – both of which require capturing then analyzing via code the spatial/temporal quantities.

Table 3: QoE models. *Grouped* - all data. *Unified* (U), S - Scene change, P - progress, SI - spatial information, TI - temporal information. *Clustered* - k-means $k = 3$ (launch), $k = 2$ (load). *Individual* - per game.

	Model	R^2 Game Launch	R^2 Level Load
	Grouped	0.60	0.65
Unified	U	0.71	0.67
	U, S	0.73	0.72
	U, P	0.71	0.71
	U, S, P	0.75	0.79
	U, SI	0.72	0.67
	U, TI	0.79	0.74
	U, SI, TI	0.81	0.82
	U, S, P, SI, TI	0.84	0.85
	Clustered	0.95 (0.03)	0.79 (0.12)
	Individual	0.96 (0.03)	0.93 (0.15)

4.4.3 Clustered models. The other row in the table labeled "clustered" is a QoE model using k-means clustering for base encoding time. We compute a silhouette score for cluster sizes from 2 to 6 for base encoding times, showing which peaks at $k = 3$ (score 0.81) for game launch and $k = 2$ (score 0.73) for level load.

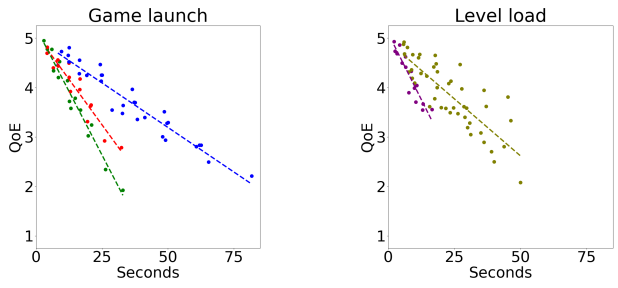

Figure 6: QoE model - Clustered

Figure 6 depicts the clustering results for game launch and level load. The x-axes are the game load times in seconds and the y-axes are the QoE. The colors represent the clusters, with the dots indicating the mean values for games in that cluster.

For game launch, the blue cluster is: GTA V, R6S, Fortnite, Civ VI, ACOD and Apex Legends. The red cluster is: LoL, Minecraft and Hearthstone. The green cluster is: PUBG, Overwatch and CS:GO. The blue, red and green regression lines through the clustered points fit the data well with R^2 0.92, 0.91, 0.97, respectively. For level load, the olive cluster is: PUBG, CS:GO, GTA V, R6S, Fortnite, LoL, Civ VI, ACOD, and Apex Legends. The purple cluster is: Overwatch, Minecraft and Hearthstone. The olive regression line fits the data moderately with R^2 0.69, while the purple regression line fits with R^2 0.87.

Compared to the grouped model, the clustered model provides the potential to more accurately predict QoE using just the base encoding time, but without needing to analyze the content nor measure quality for the individual games.

5 MEASUREMENTS

Since user QoE improves with a decrease in game load times, we designed experiments to assess how game load times vary for different hardware components – potentially helpful for game players in deciding to do an upgrade, or hardware developers as they target next generation systems. We setup a hardware testbed to facilitate adjusting configurations, implemented scripting software to automatically launch games and load their levels while measuring hardware performance, and ran repeated runs of our scripts for all games and different processors, graphics cards and storage devices.

5.1 Scripts

We designed and implemented scripts in order to automatically launch each game and load each level while recording game load times and hardware performance.

For each game, screenshots are captured for each button that the automated script needs to click in order. Then, the automated script iterates through each button screenshot, polling every 0.5 seconds until the button is displayed on the screen and pausing a minimum of 1.5 seconds between button presses. This design allows for relatively easy update to the script when a publisher's game update changes the load sequence – when this happens, the old screenshot just needs to be replaced with one or more new screenshots for the script to follow.

In cases where the loading screen can be skipped (e.g., some game cut scenes allow the player to bypass them), the scripts are set to always skip them (i.e., the game loading is done as fast as possible).

The script is written in Python. The ESP32 bot simulates all mouse presses, and PyDirectInput generates all keyboard input except for the Windows key which needs to done with PyAutoGUI. PyAutoGUI also helps detect buttons by comparing the current screen shot with the previously taken screenshot with the expected button, with a confidence threshold of 80%.

5.2 Testbed Hardware

Figure 7 is a photo of our hardware testbed. Our testbed has an open-air frame to allow for easy swapping of components and to facilitate cooling to avoid performance degredations from overheating. All configurations used: A) the all-in-one Cooler Master MasterLiquid Lite 240 evaporative water cooler; B) an EVGA 700 GD 80+ GOLD power supply; and C) Team T-Force Vulcan Z 16 GB (2x8 GB) DDR4-3000 CL16 RAM.

The testbed has two different motherboards which were swapped in as needed: the ASUS H110M-K with the LGA 1151 CPU socket

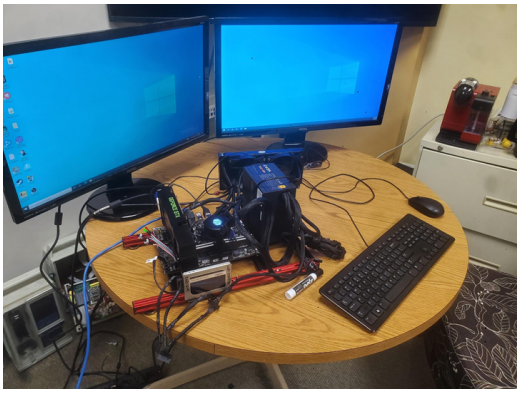

Figure 7: Hardware measurement testbed

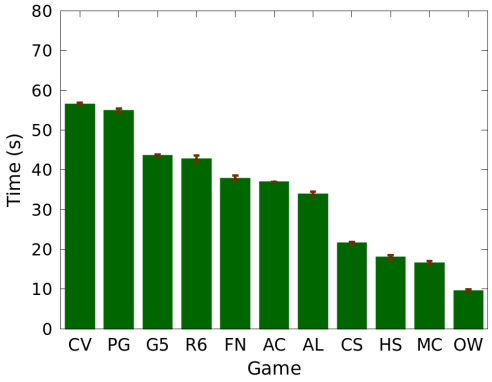

Figure 8: Game launch times

and the H410M-A with the LGA 1200 socket. The LGA 1151 socket support the 6th and 7th generation Intel processors, and the LGA 1200 the 10 generation Intel processor.

The independent variables of interest and their parameters are shown in Table 4.

Table 4: Hardware Parameters

Component	Type
CPU	Intel i5-6500 CPU @ 3.20GHz (6th Gen)
	Intel i5-7600 CPU @ 3.50GHz (7th Gen)
	Intel i5-10400 CPU @ 2.90GHz (10th Gen)
graphics	Intel CPU integrated UHD Graphics 630
	NVIDIA GeForce GTX 960
	AMD Radeon RX 5600XT
storage	Seagate BarraCuda ST2000DM001-9YN164 HDD
	Crucial MX500 500GB 3D NAND SSD

Base system: Our base system has the most recent CPU, most powerful GPU and fastest storage device in our set: 10th generation Intel i5, Crucial MX500 SSD, and the AMD Radeon graphics card. From that base system, we varied each component independently: CPU, graphics card and storage.

During game load, a script collects hardware metrics and usage data every second using Open Hardware Monitor.[3] An evaluation of the script's overhead shows in contributes less than 3% to the CPU load. All of the tests on all systems were conducted within a couple of days in March 2022 and there were no game updates between the trials on each system.

State-of-the-art system: In addition, we measured one additional hardware configuration: a ROG Strix Z590-E motherboard with an Intel i9-11900K @ 3.50GHz (11th Gen) CPU, Corsair Vengance RGB 32GB DDR4 3200 RAM, a Samsung 970 Evo Plus solid state drive, and an NVIDIA GeForce RTX 2080 graphics card. This represents a "state of the art" system in terms of our performance evaluation.

[3]https://openhardwaremonitor.org/

5.3 Data Collection

During collection, the testbed used a single monitor at 1920x1080p. The same resolution was maintained throughout all tests so that the size and location of each target button matches the pre-recorded screenshots

For level loading for multiplayer network games (e.g., League of Legends), we used single player/offline modes with bots so as not to have the load times dependent upon other player's networks.

Each game load was executed for 10 iterations in order to understand variation across runs, observe any differences in the first game load times compared to subsequent times. One round of 10 iterations took about 6 hours, after which we archive the data and swap out a single hardware component (e.g., change the graphics card), and then repeat.

Assassin's Creed level load and League of Legends game launch failed on our base system so are excluded from all level load and game launch analysis, respectively. During our measurements on our "state of the art" system, League of Legends and Assassin's Creed had been updated, breaking our scripts so those games are excluded completely from analysis on that platform only.

5.4 Measurement Analysis

This section first presents the game load times for each game on our base system, then analyzes the game load times for the different components of interest: CPU, graphics card and storage device. These latter results are also compared to our "state of the art" system.

Figure 8 and Figure 9 depict the game load times on our base system for game launch and level load, respectively. The x-axes have the games, abbreviated: Apex Legends (AL), AC Assassin's Creed: Odyssey (AC), Counter-Strike: Global Offensive (CS), Civilization VI (CV), Fortnite (FN), Grand Theft Auto 5 (G5), Hearthstone (HS), League of Legends (LoL), Minecraft (MC), Overwatch (OW), PlayerUnknown's Battle Grounds (PG), and Rainbow Six Siege (R6). The y-axis is the time in seconds. The bars are the mean game load times for the 10 iterations on the base system, ordered high to low, and shown with a 95% confidence interval.

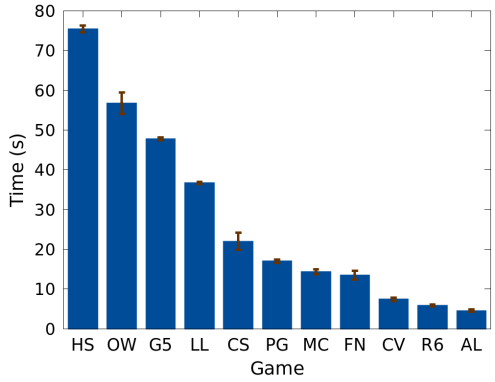

Figure 9: Level load times

Table 5: Total CPU and GPU usage percent for base system

	Game Launch		Level Load	
Game	CPU (%)	GPU (%)	CPU (%)	GPU (%)
Apex Legends	20.5	32.9	16.4	25.9
CS:GO	12.8	17.1	15.1	41.7
Civ VI	9.1	16.2	11.6	36.5
Fortnite	17.2	12.4	21.9	17.0
GTA5	8.9	15.0	8.8	20.9
Hearthstone	9.9	18.6	3.2	16.2
Minecraft	67.9	15.0	86.9	14.4
Overwatch	24.0	20.1	10.7	77.4
PUBG	18.7	28.2	41.2	81.9
R6S	15.8	12.3	14.2	29.8

Table 5 shows the overall breakdown of CPU and CPU usage recorded during game launch and level load. The units are a percentage reported by Open Hardware Monitor, averaged over all iterations. Games that load with a high CPU utilization (e.g., Minecraft) may benefit more from a more powerful CPU, while games that load with a high GPU (e.g., PUGB level load) may benefit more from a more powerful graphics card. Overall, given the sizeable CPU and GPU loads observed, we would expect CPU and GPU to both affect game load times. In addition, when the CPU and GPU are not busy, we expect the game loading to be waiting on information from storage. Thus, overall, we expect an improved storage device – an SSD versus an HDD – to improve game load times.

To assess the impact of hardware components on game load times, we measure the time it takes for game loading for one iteration of each game and then divided that by the number of games to get the average game loading performance. We do this separately for each of game launch and level load, for 9 iterations of each.

Figure 10 depicts the game loading time versus CPU generation. There are two graphs depicted: on the left is the average game launch performance and on the right is the average level load performance. The y-axes are the game load times in seconds and the x-axes are the CPU generations, ordered oldest to newest. Each bar is the mean game load time shown with a 95% confidence interval. For this and subsequent graphs, we show the game load time for our "state of the art" system on the right, separated by a bit of space and with a different color. This system has a different (better)

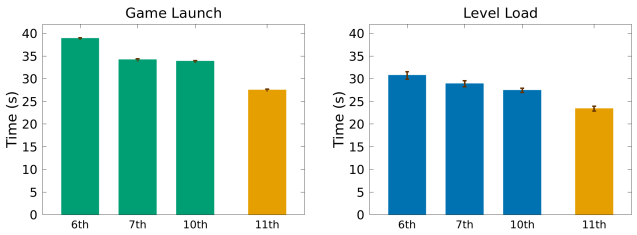

Figure 10: Game load time versus CPU generation

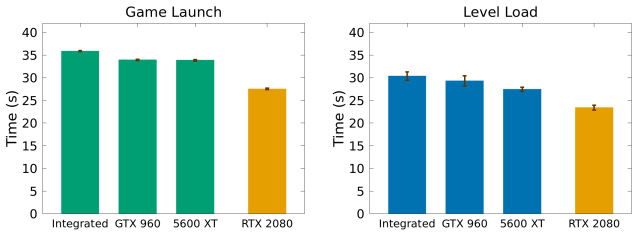

Figure 11: Game load time versus graphics card

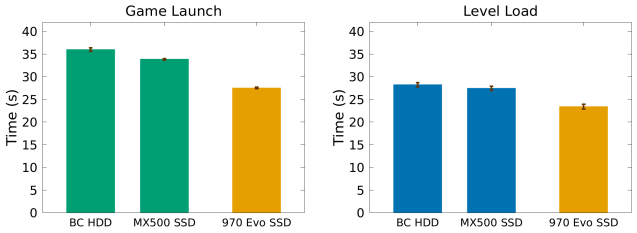

Figure 12: Game load time versus storage device type

components than our base system so is not directly comparable for the individual components, but provides a useful reference point for where our system performs relative to an overall high-end system.

From the graphs, there is a statistically significant difference in the game load times, with a reduction to average game load time for an improved CPU. Game launch times decrease about 4% per generation and level load times decrease about 3% per generation.

Figure 11 depicts the same data, but for different GPUs on the x-axes, ordered from least powerful (a GPU integrated with the CPU) to most powerful (the 5600 XT). From the graph, the GPU also makes a statistically significant difference to performance with a decrease in game load times with an increase in GPU power. Times decrease about 10% from the integrated graphics card to the best graphics card for game launch and 6% for level load.

Figure 12 depicts the same game load performance but with a different storage device a BaraCuda hard disk drive (HDD) compare to a MX500 solid state drive (SDD). The storage device also makes a statistically significant difference to performance, albeit not as much – the SDD only decreases game load times by about 3%. This is significantly less than similar measurements about a decade ago (albeit for different games) whereupon SSDs were found to provide game launch times about 25% faster than HDDs [4].

6 LIMITATIONS AND FUTURE WORK

As noted in Section 4.2.1, our user study had 54 users in total. While this sample size was large enough for statistically significant results for user quality of experience with game load time, more users may help with challenging QoE predictions for games like Apex Legends that has an individual level load linear regression R^2 of 0.46. Similarly, potentially sampling more game load time lengths, especially within the ranges we currently study, could help determine where a linear relationship does and does not hold.

Our sample is skewed towards males (only 13 females out of 54 participants). While this may reflect the gender breakdown present in some games today, the results reported may not be representative of female performance. Similarly, our study of 12 games across 7 genres was reasonably broad, yet the breadth of game genres is considerably broader. That, and the fact that games are played on a variety of devices (e.g., mobile) may mean that the QoE relationships reported here do not hold for other games and devices.

Our methodology intentionally had users watch videos of game loading instead of actual launching and play games in order to minimize human errors and ensure reproducibility of the study. While the process requires the same user interactions (e.g., mouse clicks) as an actual game loading experience, how would the results differ from actual game loading experience remains unknown.

Future work is to validate our models with more users and games and, once validated, use the models to simulate game loading experience for wide range of games. In addition, instead of linear regression, we may look to other shapes for an even better fit.

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. While users may be eager to play, waiting for game loading is inevitable and, unfortunately, the waiting time can degrade player quality of experience (QoE).

This paper presents results from a user study designed to assess player QoE under controlled game loading conditions. We recorded game loading videos and built an self-contained application to mimic the actual game loading experience. By re-encoding the game loading videos to be faster and slower, we are able to measure how game loading time directly impacts QoE. We setup a user study in a dedicated lab where 54 participants each launched and loaded 3 games (selected from 12 games) across 9 different game load times, providing subjective opinions on their experience via surveys.

Analysis of user study results shows both game launch and level load times have significant impact on player experience. Across the range of game load times studied (less than 10 seconds to over 80 seconds), a 50% decrease in game loading time improves player experience by about 0.75 points on a 5-point scale – an amount that can improve the player experience from unacceptable to acceptable. Models of QoE with game load times suggest simple linear regression can be improved by considering game load screen content or via k-means clustering based on base encoding time.

This paper also presents results from experiments that measure game load times for different hardware components of interest, including processor, graphics card and storage device. We design and implement scripts to automatically load games and record performance and run repeated iterations of loading and launching the 12 games, individually swapping out a single hardware component to assess its impact.

Analysis of results for over 48 hours of continuously launching games and loading levels shows considerable variation in game launch times and level load times, differing by 6-fold for game launch and 15-fold for level load. CPU and GPU use during game loading is similarly varied, but all games use some of each, suggesting improvements to hardware can reduce game load times. These results are born out by our analysis which shows better CPUs and graphics cards can reduce game load times by about 5% per generation, with a slightly smaller benefit to game load times (3%) for upgrading from a hard disk drive to a solid state drive.

REFERENCES

[1] Josh Allard, Andrew Roskuski, and Mark Claypool. 2020. Measuring and Modeling the Impact of Buffering and Interrupts on Streaming Video Quality of Experience. In *Proceedings of the 18th International Conference on Advances in Mobile Computing & Multimedia (MoMM)*. Chiang Mai, Thailand.

[2] Anna Bouch, Allan Kuchinsky, and Nina Bhatti. 2000. Quality is in the Eye of the Beholder: Meeting Users' Requirements for Internet Quality of Service. In *Proceedings of the ACM CHI*. The Hague, The Netherlands.

[3] Russell Branaghan and Christopher Sanchez. 2009. Feedback Preferences and Impressions of Waiting. *Human Factors* 51, 4 (2009), 528 – 538.

[4] Mark Claypool, Jared Hays, Alex Kuang, and Thomas Lextrait. 2011. On the Performance of Games using Solid State Drives. In *Proceedings of NetGames*. Ottawa, Canada.

[5] Sebastian Egger, Peter Reichl, Tobias Hoßfeld, and Raimund Schatz. 2012. 'Time is Bandwidth'? Narrowing the Gap between Subjective Time Perception and Quality of Experience. In *Proceedings of the IEEE International Conference on Communications (ICC)*. Ottawa, ON, Canada.

[6] T. Hoßfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen. 2012. Initial Delay vs. Interruptions: Between the Devil and the Deep Blue Sea. In *Proceedings of IEEE QoMEX*.

[7] Carine Lallemand and Guillaume Gronier. 2012. Enhancing User Experience During Waiting Time in HCI: Contributions of Cognitive Psychology. In *Proceedings of the Designing Interactive Systems Conference (DIS)*. Newcastle Upon Tyne, UK.

[8] Fiona Fui-Hoon Nah. 2004. A Study on Tolerable Waiting Time: How Long are Web Users Willing to Wait? *Behaviour & Information Technology* 23, 3 (2004), 153–163.

[9] Jakob Nielsen. 1993. *Usability Engineering*. Morgan Kaufmann, San Francisco, CA, USA.

[10] ITU-T Rec. 2008. Subjective Video Quality Assessment Methods for Multimedia Applications. (Accessed September 5, 2021).

[11] Dominique Scapin and J. M. Christian Bastien. 1997. Ergonomic Criteria for Evaluating the Ergonomic Quality of Interactive Systems. *Behaviour & Information Technology* 16, 4-5 (1997), 220 – 231.

[12] Wikipedia. (Accessed 2-Apr-2022). Apex Legends. https://en.wikipedia.org/wiki/Apex_Legends

[13] Wikipedia. (Accessed 2-Apr-2022). Assassin's Creed Odyssey. https://en.wikipedia.org/wiki/Assassin%27s_Creed_Odyssey

[14] Wikipedia. (Accessed 2-Apr-2022). Civilization VI. https://en.wikipedia.org/wiki/Civilization_VI

[15] Wikipedia. (Accessed 2-Apr-2022). Counter-Strike: Global Offensive. https://en.wikipedia.org/wiki/Counter-Strike:_Global_Offensive

[16] Wikipedia. (Accessed 2-Apr-2022). Fortnite. https://en.wikipedia.org/wiki/Fortnite

[17] Wikipedia. (Accessed 2-Apr-2022). Grand Theft Auto V. https://en.wikipedia.org/wiki/Grand_Theft_Auto_V

[18] Wikipedia. (Accessed 2-Apr-2022). Hearthstone. https://en.wikipedia.org/wiki/Hearthstone

[19] Wikipedia. (Accessed 2-Apr-2022). League of Legends. https://en.wikipedia.org/wiki/League_of_Legends

[20] Wikipedia. (Accessed 2-Apr-2022). Minecraft. https://en.wikipedia.org/wiki/Minecraft

[21] Wikipedia. (Accessed 2-Apr-2022). Overwatch (video game). https://en.wikipedia.org/wiki/Overwatch_(video_game)

[22] Wikipedia. (Accessed 2-Apr-2022). PUBG: Battlegrounds. https://en.wikipedia.org/wiki/PUBG:_Battlegrounds

[23] Wikipedia. (Accessed 2-Apr-2022). Tom Clancy's Rainbow Six Siege. https://en.wikipedia.org/wiki/Tom_Clancy%27s_Rainbow_Six_Siege

APPENDIX A - MODELS

$$Q_g = 0.07 \cdot b \cdot l - 0.07 \cdot l + 4.90 \quad (1)$$

$$Q_l = 0.06 \cdot b \cdot l - 0.06 \cdot l + 4.85 \quad (2)$$

Figure 13: Game launch QoE (Q_g) and level load QoE (Q_l) based on: base length (b - in seconds), actual length (l - in seconds).

$$Q_g = 0.11 \cdot b \cdot l - 0.08 \cdot l + 0.07 \cdot b + 0.003 \cdot \mathrm{SI} + 0.11 \cdot \mathrm{TI} - 0.0003 \cdot \mathrm{SI} \cdot l + 0.004 \cdot \mathrm{TI} \cdot l + 4.59 \quad (3)$$

$$Q_l = 0.10 \cdot b \cdot l - 0.09 \cdot l - 0.35 \cdot b + 0.01 \cdot \mathrm{SI} - 0.05 \cdot \mathrm{TI} - 0.0001 \cdot \mathrm{SI} \cdot l + 0.08 \cdot \mathrm{TI} \cdot l + 4.70 \quad (4)$$

Figure 14: Game launch QoE based on: base length (b - in seconds), actual length (l - in seconds), spatial information (SI - Sobel filter), and temporal information (TI- Motion difference)

$$Q_g = 0.07 \cdot b \cdot l - 0.10 \cdot l + 0.50 \cdot b + 0.11 \cdot \mathrm{p} - 0.03 \cdot \mathrm{s} + 0.002 \cdot \mathrm{p} \cdot l + 0.004 \cdot \mathrm{s} \cdot l + 4.90 \quad (5)$$

$$Q_l = 0.003 \cdot b \cdot l - 0.08 \cdot l + 0.44 \cdot b - 0.30 \cdot \mathrm{p} - 0.13 \cdot \mathrm{s} + 0.003 \cdot \mathrm{p} \cdot l + 0.01 \cdot \mathrm{s} \cdot l + 5.30 \quad (6)$$

Figure 15: Game launch QoE based on: base length (b- in seconds), actual length (l- in seconds), progressive indicator (p - 0/1), and scene changes (s - count)

$$Q_g = 0.10 \cdot b \cdot l - 0.09 \cdot l + 1.52 \cdot b - 0.004 \cdot \mathrm{SI} + 0.29 \cdot \mathrm{TI} + 0.33 \cdot \mathrm{p} - 0.12 \cdot \mathrm{s} - 0.0003 \cdot \mathrm{SI} \cdot l - 0.004 \cdot \mathrm{TI} \cdot l - 0.002 \cdot \mathrm{p} \cdot l + 0.006 \cdot \mathrm{s} \cdot l + 4.67 \quad (7)$$

$$Q_l = 0.16 \cdot b \cdot l - 0.11 \cdot l - 1.66 \cdot b + 0.01 \cdot \mathrm{SI} - 0.16 \cdot \mathrm{TI} - 0.50 \cdot \mathrm{p} + 0.05 \cdot \mathrm{s} - 0.0001 \cdot \mathrm{SI} \cdot l + 0.01 \cdot \mathrm{TI} \cdot l + 0.01 \cdot \mathrm{p} \cdot l - 0.005 \cdot \mathrm{s} \cdot l + 5.31 \quad (8)$$

Figure 16: Game launch QoE based on: base length (b- in seconds), actual length (l- in seconds), progressive indicator (p - (0/1)), scene changes (s - count), spatial information (SI - Sobel filter), and temporal information (TI - Motion difference)

APPENDIX B - MODEL GRAPHS

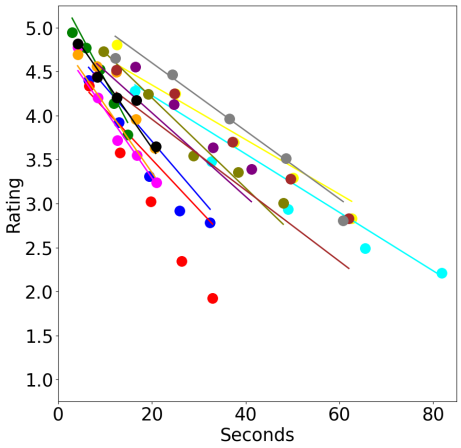

Figure 17: [U, S, P, SI, TI] model - game launch

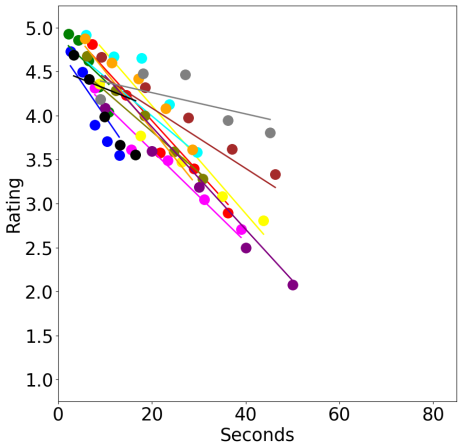

Figure 18: [U, S, P, SI, TI] model - level load

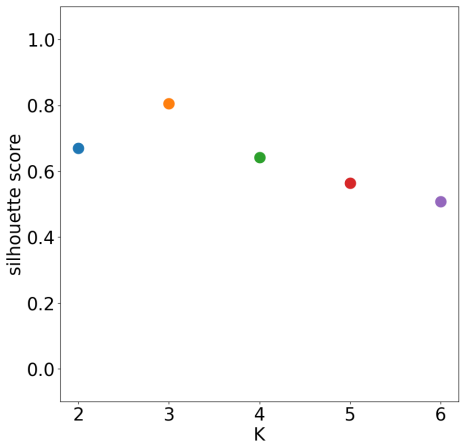

Figure 19: Average silhouette - game launch

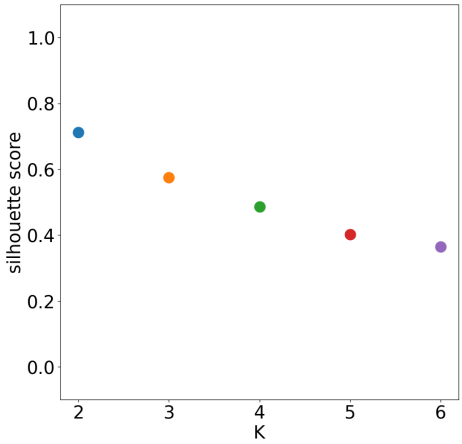

Figure 20: Average silhouette - level load

APPENDIX C - GAME LOAD TIMES PER-GAME

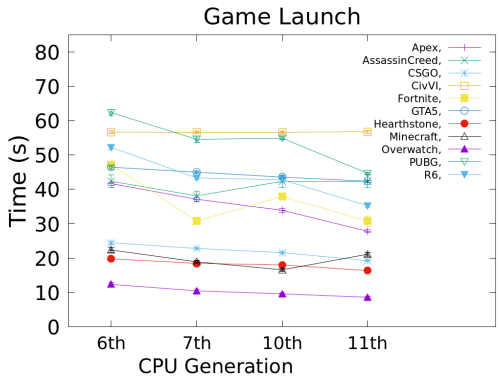

Figure 21: Per game launch time versus CPU generation

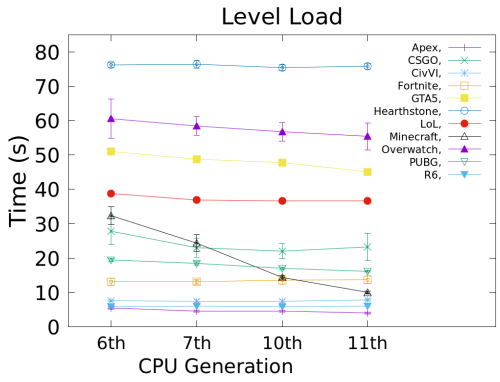

Figure 22: Per game level load versus CPU generation

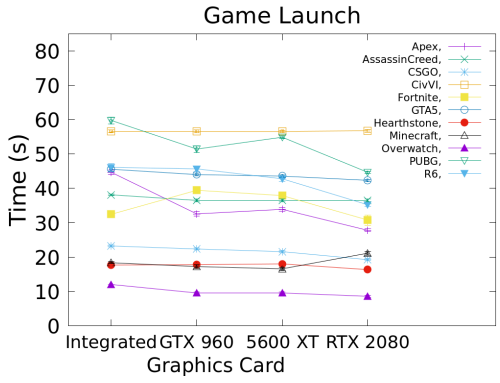

Figure 23: Per game launch time versus graphics card

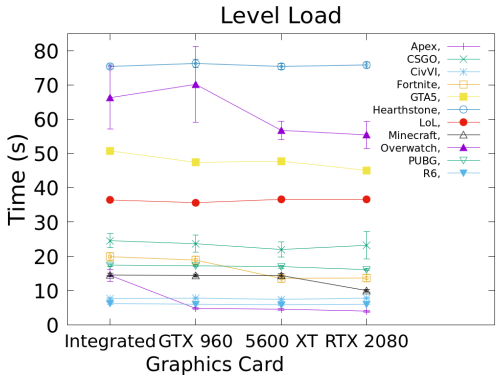

Figure 24: Per game load level time versus graphics card

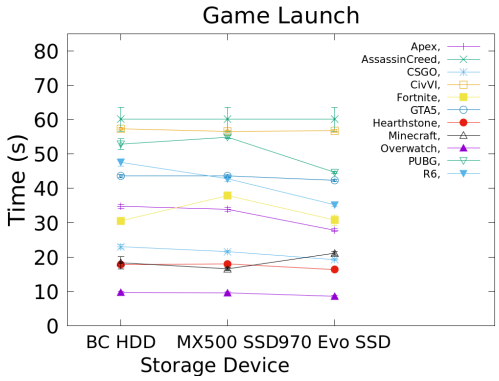

Figure 25: Per game launch time versus storage device

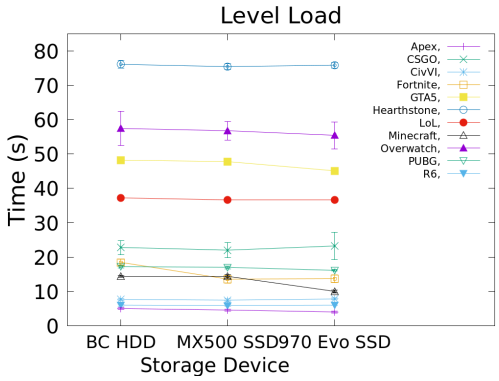

Figure 26: Per game level load time versus storage device