Leveraging AI players for QoE estimation in cloud gaming

(Article begins on next page)

Demo abstract: Leveraging AI players for QoE estimation in cloud gaming

German Sviridov*†, Cedric Beliard†, Gwendal Simon†, Andrea Bianco*, Paolo Giaccone*, Dario Rossi†
*Politecnico di Torino, Italy – first.last@polito.it
†Huawei Technologies, Co. Ltd., France – first.last@huawei.com

***Abstract*—Quality of Experience (QoE) assessment in video games is notorious for its burdensomeness. Employing human subjects to understand network impact on the perceived gaming QoE presents major drawbacks in terms of resources requirement, results interpretability and poor transferability across different games.**

To overcome these shortcomings, we propose to substitute human players with artificial agents trained with state-of-the-art Deep Reinforcement Learning techniques. Equivalently to traditional QoE assessment, we measure the in-game score achieved by an artificial agent for the game of Doom for varying network parameters. Our results show that the proposed methodology can be applied to understand fine-grained impact of network conditions on gaming experience while opening a lot of new opportunities for network operators and game developers.

I. Introduction

Cloud gaming is ready to take off. Playing a game at home while the game engine runs in the cloud and the game scene is streamed over regular Internet has become a reality, which leverages a decade of research effort [1]. Multiple companies have recently released their own cloud gaming services, including online giants,[1] hardware vendors,[2] game studios,[3] console manufacturers, [4] and game engine vendors.[5]

While the technologies are mature to run cloud gaming services, offering an excellent Quality of Experience (QoE) comes at the cost of huge expenses: (i) The requirements of game engines often include access to expensive servers with GPU and multiple cores; (ii) The displayed scene is extracted from a video stream. The more bit-rate for streaming, the higher visual quality, but the more expensive; (iii) The lag (delay between an action triggered by the gamers and the impact of this action on their screen) must be reduced, which calls for distributed solutions with engine placement optimization [2]. Yet, the reservation of high-performance servers close to the end-users means cost overhead.

Fortunately, the literature on QoE assessment of games in delayed environment has opened the door for optimization. Not all the games are equal: the gamers may barely feel a 150 ms lag in one game although they regard another game as being un-playable with a lag above 80 ms [3]. The latter should run in priority at the edge while the former may be opportunistically placed in further cheaper servers. Unfortunately, it is hard to characterize the lag sensitivity of a game without extensive subjective tests.

In this demo, we show our approach for automatizing the evaluation of game sensitivity to network disturbance, in particular latency, packet loss, and jitter. We leverage recent advances in the field of Artificial Intelligence (AI) for autonomous learning gamers [4]. This breakthrough provides a versatile tool for characterizing the sensitivity of a game by employing artificial gamers and confronting them to various network QoS. We foresee a significant improvement in the process of ingesting a new game in a game catalog. For the cloud gaming service provider, the consequence is an opportunity for better management and substantial cost savings. In this demo we provide a showcase of game QoE assessment by employing artificial agents that are built using Deep Reinforcement Learning (DRL) technique instead of human gamers and by evaluating their performance while we control the network QoS parameters.

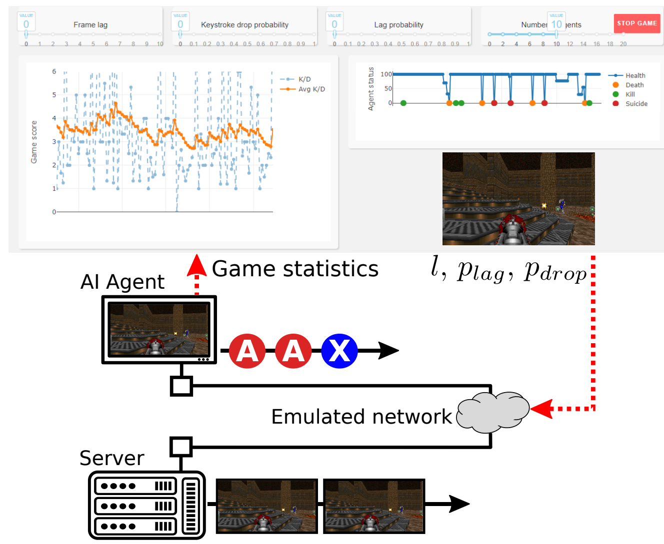


Fig. 1: Cloud Gaming (CG) emulated system.

II. Methodology

The proposed methodology consists in training an AI agent and letting it play in an emulated Cloud Gaming (CG) network environment. During the play-through we alter the network

[1]Google Stadia – https://stadia.google.com/
[2]Nvidia GeForce Now – https://www.nvidia.com/en-us/geforce-now/
[3]Gameloft Blacknut – https://www.blacknut.com/assets/press-release/cp-gameloft-blacknut.pdf
[4]Sony – https://www.playstation.com/en-us/explore/playstation-now/
[5]Unreal Engine Pixel Streaming – https://docs.unrealengine.com/en-US/Platforms/PixelStreaming/index.html

TABLE I: Network QoS conditions in the demo

Variable	Description	Range
p_{drop}	Per-keystroke drop probability	[0, 0.5]
p_{lag}	Per-frame lag probability	[0, 0.8]
l	Lag duration	[0, 300] ms

conditions and measure the agents gaming performance under new conditions. An overview of the used architecture, alongside with a demo example is depicted in Fig. 1.

A. Artificial agent and in-game score

On the client side we employ an artificial agent trained to play Doom. The agent is trained using the deep neural network architecture described in [5] which won the first place in the yearly ViZDoom competition [6]. Identically to a human player, the agent receives as input raw pixels and based on it takes decisions in the form of keystrokes.

We employ the average in-game score as an indicator of the perceived QoE as previously done in [7]. We consider the kill over death ratio (K/D) as the primary QoE indicator since it captures various in-game statistics (including accuracy, avoidance, and reactiveness) and more generally reflects how easy it is to play the game in a given configuration.

B. Cloud Gaming (CG) architecture

On the server side we run the actual game engine, which computes the game state in reaction to the actions received from the agent. The server sends a continuous stream of frames to the client at the same frame-rate as the game engine (35 frames per second). We do not implement complex video encoding since we do not deal with throughput in this demo.

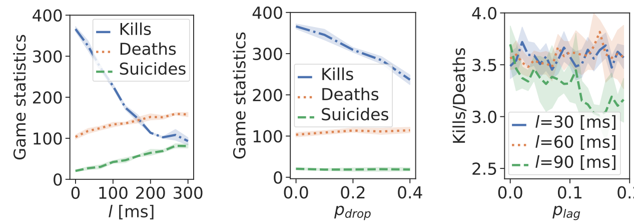
Both the agents and the server are in the same high-performance network; we are thus able to measure the gaming experience (i.e., AI agent score) in both ideal conditions (sub-millisecond delay, high-bandwidth, no loss) as well as under controlled network conditions. We employ parameters depicted in Table I to set the network conditions. We perform a per-keystroke drop with probability p_{drop}, and add a fixed delay l with probability p_{lag} to each frame transmitted from the server to the client.

III. Demo highlights

The demo offers an interactive showcase involving an artificial agent playing Doom against the in-game bots.

Demo setup. Each agent continuously sends game statistics to the demo front-end, thus permitting to track its in-game performance in real-time. The number of agents concurrently playing can be adjusted with respect to the underlying hardware resources. The demo back-end runs on a server equipped with an Intel Xeon E5-4627 CPU, which supports ten concurrent game sessions without glitches in real-time result collection.

Demo Workflow. The users are allowed to interact with the demo by adjusting network parameters from Table I and observing in real-time how the performance of the agent varies. Notably users are given the opportunity of acting upon the parameters shown in Table I. As the result of network conditions variations, it is possible to observe the in-game score variation alongside with additional selected in-game statistics. To have a visual glance on how agents react to network conditions, users are provided with the actual gameplay visualization of one of the agents. We also provide a very brief video of the demo is available online [8].



(a) Deterministic lag (b) Stochastic losses (c) Stochastic lag

Fig. 2: Score degradation in perturbed channel scenario

Demo scenarios. We consider a game scenario with different types of deathmatches (i.e., all-vs-all competition), which are characterized by the complexity of the map and availability of in-game weapons. Fig. 2 depicts experimental results alongside with the 95 % confidence intervals after multiple deathmatch runs for a subset of selected network parameters. Our demo permits to obtain similar results in real-time for any combination of network parameters.

IV. Conclusion

In this demo we present an innovative methodology of estimating gaming QoE by leveraging artificial players trained with deep learning techniques instead of employing real human subjects. Our results give insights on the possible applications of the proposed methodology in the field of flow scheduling and resource orchestration.

References

[1] W. Cai, R. Shea, C.-Y. Huang, K.-T. Chen, J. Liu, V. C. Leung, and C.-H. Hsu, "A Survey on Cloud Gaming: Future of Computer Games," *IEEE Access*, 2016.

[2] S. Choy, B. Wong, G. Simon, and C. Rosenberg, "A Hybrid Edge-Cloud Architecture for Reducing On-Demand Gaming Latency," *Multimedia systems*, 2014.

[3] M. Claypool and D. Finkel, "The Effects of Latency on Player Performance in Cloud-based Games," in *IEEE/ACM Workshop on Network and System Support for Games (NetGames)*, 2014.

[4] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, 2015.

[5] G. Lample and D. S. Chaplot, "Playing FPS Games with Deep Reinforcement Learning," in *Conf. on Artificial Intelligence (AAAI)*, 2017.

[6] M. Wydmuch, M. Kempka, and W. Jaśkowski, "Vizdoom competitions: playing doom from pixels," *IEEE Transactions on Games*, 2018.

[7] R. Amin, F. Jackson, J. E. Gilbert, J. Martin, and T. Shaw, "Assessing the Impact of Latency and Jitter on the Perceived Quality of Call of Duty Modern Warfare 2," in *Int. Conf. on Human-Comp. Inter. (HCI)*, 2013.

[8] https://www.dropbox.com/s/uhnf67ipvnwpucd/QoEdemo.mp4.