DEMO: CGSynth: Cloud Gaming Synthesizer

Alireza Shirmarz
UFSCar, Brazil

Ariel G. de Castro
UNICAMP, Brazil

Victor H. S. Lopes
UNIPAMPA, Brazil

Fabio Verdi
UFSCar, Brazil

Marcelo C. Luizelli
UNIPAMPA, Brazil

Christian Rothenberg
UNICAMP, Brazil

ABSTRACT

Cloud gaming's unique network traffic is challenging to reproduce for research. This demo introduces Cloud Gaming Synthesizer (CGSynth), a platform that generates realistic, configurable synthetic cloud gaming (CG) traffic. CGSynth captures real CG patterns and allows their synthetic reproduction with user-defined flow/packet parameters and deterministic protocol headers. It employs an LSTM for accurate, order-preserving timestamp generation and AI-based video interpolation for realistic payloads. Crucially, CGSynth integrates a QoE evaluation module using objective (e.g., SSIM) and subjective metrics (e.g., MOS) to validate synthetic traffic's video quality and responsiveness against real streams.

1 INTRODUCTION

Cloud Gaming (CG) is rapidly becoming a mainstream content delivery model, making robust quality-of-experience (QoE) assessment – both objective and subjective – a key research and industry priority [1, 4]. However, rigorous scientific progress in CG is persistently hindered by two main challenges: (*i*) lack of experimental reproducibility across different gaming platforms, and (*ii*) difficulty in systematically manipulating rendered frames and network scenarios.

The opaque nature of commercial CG platforms (e.g., Xbox Cloud Gaming, GeForce Now) restricts access to internal metrics and control, complicating controlled QoE evaluation. Furthermore, the inherent nondeterminism of gameplay, where identical inputs may not produce identical outputs, makes direct session comparison and analysis problematic.

To overcome these obstacles in reproducible and configurable CG traffic generation, we introduce CGSynth, a modular and extensible framework. CGSynth builds upon CGReplay[1] [8], significantly extending its capabilities beyond simple traffic replay. Our system distinctively supports: *(i)* fine-grained synthetic traffic configuration, including protocol headers (e.g., Ethernet, IP, UDP); *(ii)* realistic timestamp generation via a Long Short-Term Memory (LSTM) model trained on real CG PCAP; and *(iii)* comprehensive traffic packetization that synthesizes complete packets, integrating timestamps, headers, and AI-generated payloads.

[1]https://github.com/dcomp-leris/CGReplay

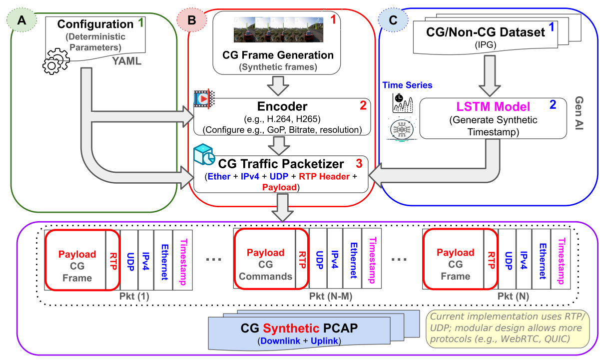

Figure 1: CGSynth architecture modules: (A) System Configuration, (B) Packet Synthesizer, and (C) Synthetic Trace Generator.

Synthetic downlink payloads are generated by adaptively manipulating CG video frames – such as introducing compression artifacts or frame drops – while preserving original game-specific visual characteristics. While uplink player command traffic is derived directly from CGReplay, the corresponding downlink video stream is entirely synthesized by CGSynth. Both traffic streams are carefully synchronized to maintain temporal fidelity with real CG sessions.

To validate the perceptual quality of the synthesized traffic, CGSynth incorporates a robust evaluation framework. We compare synthetic and real downlink video streams using both objective computational metrics (e.g., SSIM, LPIPS) and subjective user feedback gathered via a web-based QoE assessment platform. Furthermore, the realism of the synthetic downlink video is enhanced using AI-based deep frame interpolation techniques (e.g., RIFE[5]), ensuring consistently high perceptual similarity to original gameplay footage.

This demonstration showcases CGSynth as a complete pipeline for reproducible and exploratory interactive CG experimentation. Our framework is designed to enable new insights into the complex interplay between network conditions, CG system performance, and end-user experience.

2 ARCHITECTURE & DEMO

The CGSynth architecture aims at a modular platform meticulously designed for reproducible cloud gaming experimentation. While CGReplay focuses on capturing and replaying real-world CG traffic (uplink/downlink), CGSynth enhances this by generating versatile synthetic CG traffic, culminating in complete PCAP files with all pertinent packets. As depicted in Figure 1, CGSynth comprises three core integrated components (A, B, and C), comprehensively detailed below.

(A) System Configuration. The configuration is YAML-file based to provide flexible and reproducible control over traffic generation and network conditions. Users can define parameters like the number of flows, packet sizes, and header fields (e.g., Ethernet, IP, UDP), with optional randomization for realism. It also supports synchronization inputs from CGReplay to preserve the timing and order of commands and frames. Network settings such as bandwidth, delay, loss rates, and queue management can be configured for both software (Mininet-WiFi [3]) and hardware (P4/Tofino) setups.

(B) Packet Synthesizer. This module serves as the core integration point. It assembles complete cloud gaming packets using three main inputs: synthetic timestamps from Module C, deterministic headers (Ethernet, IP, UDP) from the configuration, and payloads representing gaming content. Uplink commands are reused directly from CGReplay, while downlink video frames are generated synthetically to match the original gameplay. Downlink generation follows three steps: (1) new frames are synthesized using CGReplay input and interpolation methods like AI-based RIFE or OpenCV's addWeighted, (2) frames are compressed using standard encoding to simulate streaming, and (3) packets are created in sync with the original command-frame order. This ensures accurate interleaving of commands and video data, preserving realistic session behavior. The module also supports resolution scaling, frame degradation, and batch processing. For QoE assessment, it includes both objective (PSNR, SSIM) and perceptual (LPIPS [9], VMAF [7]) metrics, enabling detailed evaluation of visual quality and user experience [6].

(C) Synthetic Trace Generator. This module generates realistic synthetic timestamps by analyzing inter-packet gaps (IPGs) from captured PCAPs. It trains an LSTM model on IPG data extracted from game replays (e.g., Fortnite), learning the temporal patterns between consecutive packets. Once trained, the model predicts future IPG values used to compute synthetic timestamps. These timestamps are then passed to Module B for precise orchestration of RTP packet generation, ensuring high temporal fidelity in the synthetic traffic.

These three modules work together as part of a unified experimental pipeline. Module A handles overall configuration, such as network settings and packet structure. Module C generates synthetic timestamps while preserving the original order of commands and downlink frames to maintain high fidelity. Module B uses this information to create realistic downlink payloads based on game content, mixes them with the commands, and builds complete packets according to the setup defined in Module A.

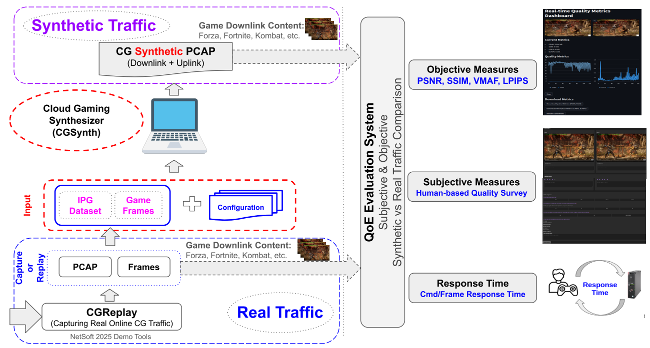

Figure 2: Demonstration setup.

During the demonstration. We will showcase the full system pipeline using prerecorded PCAPs and video frames captured with CGReplay from three popular games: Forza, Fortnite, and Mortal Kombat (Figure 2). The captured traffic is used as input to generate synthetic CG traffic with our CGSynth platform. The synthetic traffic is then fed into our QoE evaluation system to assess both subjective and objective quality, as well as response time. Participants will observe and compare the QoE between real CGReplay traffic and the corresponding synthetic traffic, demonstrating the effectiveness and reproducibility of our framework.

3 CONCLUSIONS & FUTURE WORK

The CGSynth demonstration presents a novel, modular framework for reproducible, controllable cloud gaming experimentation. CGSynth enables synthetic video generation, network-aware replay, and comprehensive QoE assessment (objective and subjective). With AI-based video interpolation and PCAP generation, it allows a systematic investigation of how visual/network impairments affect user experience.

The demonstration shows the ability of CGReplay to combine synthetic video frames with emulated networks for in-depth QoE analysis. By correlating subjective user feedback with objective metrics (e.g., SSIM, LPIPS), we highlight how CGSynth bridges the gap between technical measurements and perceptual quality across diverse gaming scenarios.

Future work includes upstream input modeling, modern codec support (e.g., AV1), more protocols (e.g., WebRTC, RTP over QUIC (RoQ) [2]), real-time QoE inference, and AI-driven frame enhancements like super sampling (e.g., DLSS), potentially triggered by network conditions or gaming contexts. We also plan a public dataset of synthetic and real traces with QoE annotations to foster community research.

REFERENCES

[1] Gergely Dobreff, Dániel Frey, Attila Báder, and Alija Pašić. 2024. Empowering isps with cloud gaming user experience modeling: A nvidia

geforce now use-case. In *2024 27th Conference on Innovation in Clouds, Internet and Networks (ICIN)*. IEEE, 202–209.

[2] M. Engelbart et al. 2025. *RTP over QUIC (RoQ)*. Internet-Draft draft-ietf-avtcore-rtp-over-quic-13. Internet Engineering Task Force. https://datatracker.ietf.org/doc/draft-ietf-avtcore-rtp-over-quic/13/ Work in Progress.

[3] Ramon R Fontes, Samira Afzal, Samuel HB Brito, Mateus AS Santos, and Christian Esteve Rothenberg. 2015. Mininet-WiFi: Emulating software-defined wireless networks. In *2015 11th International conference on network and service management (CNSM)*. IEEE, 384–389.

[4] Jin Heo, Ketan Bhardwaj, and Ada Gavrilovska. 2024. Adrenaline: Adaptive Rendering Optimization System for Scalable Cloud Gaming. *arXiv preprint arXiv:2412.19446* (2024).

[5] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. 2022. Real-Time Intermediate Flow Estimation for Video Frame Interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[6] Georgios Kougioumtzidis, Atanas Vlahov, Vladimir K. Poulkov, Pavlos I. Lazaridis, and Zaharias D. Zaharis. 2024. QoE Prediction for Gaming Video Streaming in O-RAN Using Convolutional Neural Networks. *IEEE Open Journal of the Communications Society* 5 (2024), 1167–1181. https://doi.org/10.1109/OJCOMS.2024.3362275

[7] Zhi Li, Christos Bampis, Julie Novak, Anne Aaron, Kyle Swanson, Anush Moorthy, and Jan De Cock. 2018. VMAF: The Journey Continues. https://netflixtechblog.com/vmaf-the-journey-continues-44b51ee9ed12. (2018). Accessed: 2025-05-17.

[8] Alireza Shirmarz, A. G. de Castro, F. L. Verdi Verdi, and C. Rothenberg. 2025. CGReplay: Capture and Replay of Cloud Gaming Traffic for QoE/QoS Assessment. In *2025 IEEE 11th International Conference on Network Softwarization (NetSoft)*. https://netsoft2025.ieee-netsoft.org/

[9] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.

4 TECHNICAL REQUIREMENTS

4.1 Equipment to be used for the demo

A single laptop.

4.2 Space needed

The space provided by SIGCOMM.

4.3 Setup time required

Up to 30 minutes.

4.4 Additional facilities needed, including power and any Internet access requirements

Internet access and a power source for a laptop are required.

5 DOCUMENTATION AND CODE

CGSynth is released as an open-source project under Apache-2.0 license. It can be downloaded from Github at:

.