A Cost-Efficient Cloud Gaming System at Scale

Yiling Xu, Qiu Shen, Xin Li, and Zhan Ma

Abstract

This article proposes a transparent gaming (TG) cloud system that allows users to play any popular high-end desktop game on the fly over the Internet. Toward this goal, we have introduced the TG-SHARE technology to share the underlying hardware capabilities, particularly for the GPU and the dedicated compression acceleration unit (XCODER). TG-SHARE utilizes off-the-shelf consumer GPUs without resorting to expensive proprietary GPU virtualization technology (e.g., GRID from NVIDIA). XCODER adapts the compression based on the network dynamics, learned gaming behaviors, and hardware resources to significantly reduce bandwidth consumption. Google's webRTC protocol is integrated to offer real-time interaction and ubiquitous access from heterogeneous devices. Compared to the existing cloud gaming vendor using the GRID technology, our TG-SHARE not only reduces the expense per user (i.e., 75 percent hardware cost reduction, 20–40 percent network cost reduction), but also improves the quality of experience with higher rate of frames per second (i.e., 2× FPS).

Introduction

Video gaming is a multi-billion-dollar business with compound annual growth rate of 6.6 percent in the coming years [1]. Games can be accessed through consoles (e.g., Xbox, Play Station) and mobile devices as well as personal computers (PCs). Oftentimes, games rendered on consoles and PCs offer better visual quality and user experience because of their superior processing power, while games on mobile handhelds provide a pervasive gaming experience, but have to sacrifice a certain degree of quality due to energy and thermal concerns.

Users expect a way to support both mobility and quality, while game producers also prefer a unified platform that can make the contents accessible to heterogeneous users. *Cloud gaming* is an effective means that enables such ubiquitous quality. Games are rendered at cloud servers with instantaneous scenes compressed and delivered as streaming video to users, while the other end captures the user inputs (key strokes, mouse clicks, joystick movements, etc.) and sends them back to the cloud for interaction.

Efforts have been devoted to cloud gaming since the 2000s, resulting in some well-known startups, including G-cluster, OnLive, Gaikai (now Sony PS NOW), NVIDIA GeForce NOW, the more recent LiquidSky, and others [2]. Research has expanded from architecture design to compression optimization to network planning to user experience study and so on [3]. Unfortunately, they have not reached a very successful and practical model so far. Three players are in this business: the game provider, the end user, and the technology vendors (e.g., the aforementioned startups). We believe that the technology vendors aim to offer cloud gaming services that should be *transparent* and *cost-efficient* to both the game producers and the end users for potential success [4].

In this article, we have developed a transparent gaming (TG) cloud, TG-cloud, shown in Fig. 1a. It is yet another practical application of *transparent computing* [4–6] in reality. More specifically, TG-cloud consists of massively geo-distributed gaming data centers (DCs). Each DC includes hundreds or thousands of interconnected TG-nodes. Usually, portal nodes, shown in Figs. 1b and 1c, are installed to host both the user and the resource manifests. A TG-node, depicted in Fig. 1d, is typically a gaming server with one or two high-end CPUs, one or more powerful graphic processing units (GPUs), multi-channel compression acceleration (i.e., XCODER), memory (RAM), disk, network (NET), and other components. For each TG-node, we have implemented a TG-SHARE strategy to run multiple gaming client applications simultaneously to share the underlying hardware resources. The gaming client of any popular desktop game can be installed as an isolated application (app) without requiring additional modifications from the game provider. Instantaneously rendered gaming scenes are compressed and streamed to the users over the Internet, while the users' interactions from the keyboards, mice, and joysticks are sent back to the server in real time. We utilize the network dynamics, learned gaming behaviors, and hardware resources to adapt the compression so as to significantly reduce bandwidth consumption.

Transparent gaming is achieved without requiring any additional efforts from either the game providers (e.g., Tencent or Blizzard) or the users. Users only need to download and install the game clients on our TG-cloud as they have done before on their local PCs. They can then play the games on the fly anywhere remotely. The GPU virtualization and network bandwidth consumption are two major costs for cloud gaming. Herein, the vGPU technology is usually enabled by those professional and expensive GPUs. Besides, software licensing is also required for its enabling. On the contrary, our TG-SHARE strategy could

Digital Object Identifier:
10.1109/MNET.2018.1700153

Yiling Xu is with Shanghai Jiaotong University; Qiu Shen and Zhan Ma are with Nanjing University; Xin Li is with Yun Ge Zhi Li Inc..

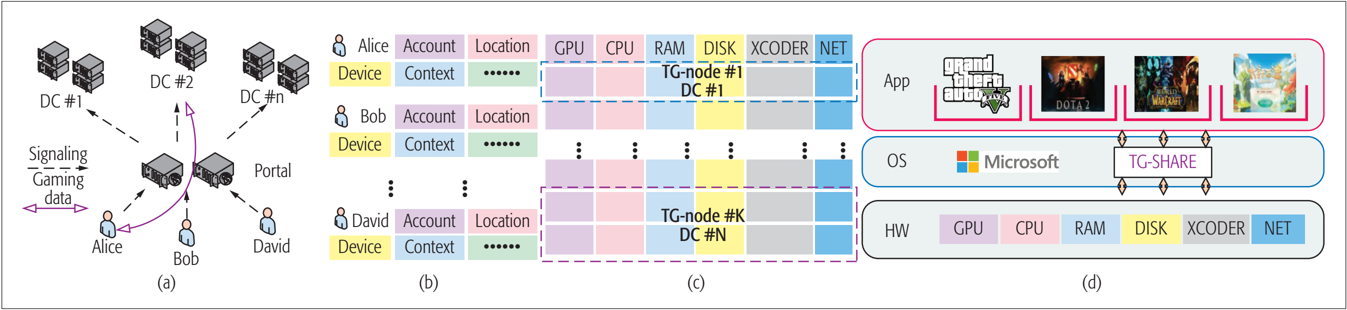


FIGURE 1. Illustration of the Transparent Gaming architecture: a) TG-cloud (b) user manifest; c) resource manifest; d) TG-node with TG-SHARE strategy.

leverage low-cost consumer GPUs and does not need to pay the extra licensing fees of the existing vGPU technologies. Meanwhile, the dedicated compression acceleration unit (XCODER) could save network bandwidth significantly, so the network expenses of both cloud gaming vendors (i.e., uplink) and users (i.e., downlink) can be reduced. Our extensive field tests have shown that TG-SHARE could reduce the hardware cost about 75 percent and network expense about 20–40 percent, and also improve the gaming frames per second by a factor of 2 in comparison to the GRID platform.

The remainder of this article is organized as follows. We detail the architecture and implementations of the proposed TG-cloud, particularly for the TG-SHARE strategy, followed by a performance study and comparison to GRID-based cloud gaming. We then conclude the work.

Transparent Gaming: Architecture and Implementation

This section describes the details of our TG-cloud system as shown in Fig. 1. Both users and TG-nodes need to register with the corresponding manifest database hosted in portal nodes. Whenever a new user registers, or a new TG-node is included in the system, its associated information (Fig. 1c) is appended elastically into the current manifest database. Upon an instantaneous gaming request from a specific user, an appropriate TG-node with sufficient resource will be chosen from a proper DC according to the user's context (account, location, device, gaming behavior, etc.). The TG-node is the basic core unit in the TG-cloud that hosts gaming clients and interacts with users remotely. The following paragraphs detail the TG-node implementation.

As aforementioned, a TG-node is usually a gaming server, and multiple games run on it simultaneously. Multiple games can be accessed by multiple users remotely. Such a multi-user scheme on a single TG-node can be realized using a virtualized machine (VM) or an application container to share the underlying hardware resources. Herein, GPU sharing or virtualization is the key issue for cloud gaming. Most of the existing cloud gaming vendors, such as gCloud (www.gcloud.cn), directly use the proprietary GRID technology from NVIDIA to provide vGPUs [7]. For each vGPU, dedicated resources, such as CUDA cores and hardware encoding acceleration, are allocated to a specific game.

Dedicated GPU pass-through is an alternative way, by which a standalone GPU chip is dedicated to a single user. However, GPU pass-through limits the number of users simultaneously using a TG-node. This is because the number of PCI-E interfaces and the number of sizable GPU devices are both limited for a standard rack server.

Instead of licensing the existing vGPU technologies or applying the GPU pass-through, we have developed our own resource sharing strategy — transparent gaming share (TG-SHARE) — to allow remote users to play games in parallel. As illustrated in Fig. 2, there are three core functions defined in TG-SHARE to enable resource (particularly GPU) sharing and remote access.

System Sensing: SysSENSE

The **SysSENSE** module performs the system sensing to periodically monitor the available sources from the underlying hardware. Upon a request from a user, this subsystem coordinates with the portal nodes to balance the overall load for the TG-cloud.

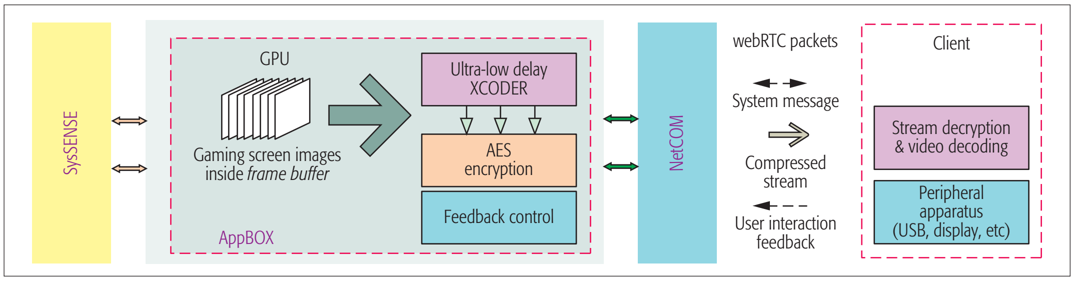
Application Box: AppBOX

The **AppBOX** subsystem is an application container where each individual game is isolated using sandboxie (https://www.sandboxie.com/). Hardware resources, such as CPU, GPU, and XCODER, are allocated to the application contained in the AppBOX. More specifically, for each game, its instantaneous graphical user interface (GUI) image scenes are actively fetched from the GPU frame buffer and compressed using the dedicated XCODER.

In comparison to the GRID technology, which requires professional GPU cards, our solution supports the multi-user scenario using off-the-shelf consumer-level GPUs. For example, we can use the consumer-grade GTX1080 GPU that costs less than US$700 compared to the professional-grade Telsa M60 that costs more than US$3000. While costing four times more than the GTX1080, the M60 could not support four times more users than the GTX1080 for high-end games according to our simulations. Thus, GTX1080 is more cost effective per user than M60.

XCODER is a subsystem to manage the ultra low delay encoding or transcoding adaptively. It can be realized in software fashion running on a CPU. It can also use the hardware codec engine in a GPU (e.g, NVENC and NVDEC in NVIDIA's products). Moreover, it can be a dedicated hardware acceleration (HWA) chipset.

XCODER will choose the most appropriate means (i.e., CPU, GPU, or dedicated HWA) to



SysSENSE
GPU
Gaming screen images inside frame buffer
Ultra-low delay XCODER
AES encryption
Feedback control
AppBOX
NetCOM
webRTC packets
System message
Compressed stream
User interaction feedback
Client
Stream decryption & video decoding
Peripheral apparatus (USB, display, etc)

FIGURE 2. Illustration of the core functions of the TG-SHARE scheme.

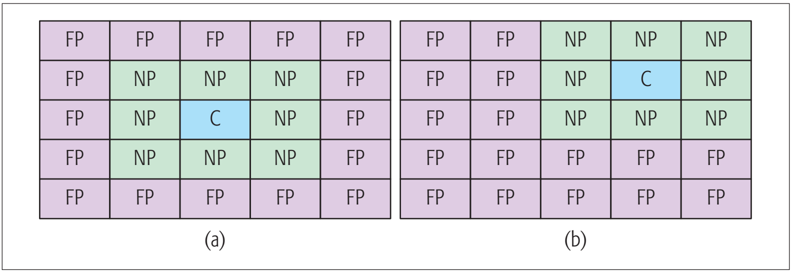


FIGURE 3. Unequal compression using the HVS characteristics and gaming behavior: a) initial 5 × 5 pattern at *t*; b) gaming interaction induced central area movement at *t* + 1.

accelerate the compression depending on the delay (i.e., from the application, system and network sensing via the NetCOM interface) and available resources. For example, gaming scenes are typically rendered using the GPU. The most straightforward way is using the codec engine within the GPU. But the number of codecs supported by a consumer-level GPU is limited. For instance, only two NVENCs are available in a GTX1080 card. Alternatively, frames can be encoded using a CPU or dedicated HWA, but obviously extra delay is incurred because of the need to transfer raw data outside the GPU. Raw frames can be lightly compressed in an ultrafast manner to reduce the data size and perform the transcoding on a CPU or dedicated HWA. Software (SW) in either a CPU or dedicated HWA offers additional options that have not been supported by the GPU, including alternative compression options, advanced encoding adaptation, and so on.

To ensure pervasive access, compression methods utilized in XCODER are compliant with the popular video coding standards, such as H.264/AVC [8] and high-efficiency video coding (HEVC) [9]. Thus, the baseline performance of compression is mainly dependent on codec implementation and optimization. Gaming experience is very subjective. Thus, instead of optimizing performance toward objective measurements, such as mean squared error, we focus more on the perceptual quality enhancement.

Hence, we introduce gaming behavior learning to further reduce network bandwidth consumption while preserving the same subjective quality. This is motivated by the fact that a user is extraordinarily focused when playing a game. Instead of applying uniform quality for whole gaming scenes, we can utilize the unequal vision impacts [10] of the central and peripheral areas by assigning different compression factors. As shown in Fig. 3, a 5 × 5 tiling pattern is used initially with three sub-areas aligned with the human visual system (HVS) [10], for example, the central (C) area, and the near peripheral (NP) and far peripheral (FP) areas. Our studies have shown that we could reduce the quality in the NP and FP areas by setting different quantization parameters (QPs), that is, $QP_{NP} = QP_C + 6$, $QP_{FP} = QP_{NP} + 6$, without perceiving overall quality degradation subjectively. With such unequal QP assignments, compression bit rate will be reduced noticeably by 20 to 40 percent. Reduction variations are content-dependent. The C area will move along with the gaming interactions. This can be captured by the mouse or joystick movements via the feedback channel. Particularly for mobile devices, we could utilize mobile crowdsouring techniques [6] to better explore gaming behaviors and further improve user experience.

Network Communication: NetCOM

The **NetCOM** part encapsulates the compressed stream into webRTC (https://webrtc.org) compatible packets and delivers them to the remote users. Meanwhile, users' feedback, such as mouse clicks, keyboard typing, and joystick movements, is collected and sent back through the NetCOM to fulfill the end-to-end interaction. Google's webRTC is chosen as the transport protocol to ensure ultra low-delay interaction and transparent access through either web browsers or compatible software applications from heterogeneous user terminals.[1]

The webRTC framework integrates the Google Congestion Control (GCC) algorithm to proactively sense the network dynamics. For instance, GCC employs two controllers: a sender-side controller, which computes the target sending bit rate A_s, and a receiver-side controller, which computes the bit rate A_r that is returned to the sender. The sender will adjust the sending bit rate to $\min(A_s, A_r)$. The sender-side controller estimates the sending rate according to the packet loss rate reported periodically by the client. The receiver-side controller is a delay-based congestion control algorithm based on the packet arrival time estimation. We scale the estimated sending bit rate by 0.9 as the compression bit rate $R = 0.9 \times \min(A_s, A_r)$ for the underlying XCODER to perform the network adaptive encoding. Furthermore, we could leverage deep learning techniques to control network traffic intelligently [11, 12].

Performance Study and Comparison

To understand the gaming experience of the proposed TG-cloud well, we perform experiments by collecting actual gaming data in real life. Toward this goal, we build a small private TG-cloud

[1] This is because most leading technology giants (e.g., Google, Microsoft, Apple) support the webRTC protocol.

at Chengdu, one of several famous game content production centers in China. This TG-cloud includes 10 TG-nodes, each equipped with 32G RAM, 512G SSD, 8G GTX1080 GPU, 8-core Xeon CPUs, and a self-designed hardware chip offering 16-channel H.264/AVC [8] and HEVC [9] compression acceleration simultaneously. Servers are interconnected using 1000 Mb/s Ethernet.

TG-cloud is hosted by a family type Internet service provider (ISP) offering constant 50 Mb/s uplink. As shown in Fig. 5, unified access (http://www.gwecom.com:18080/vapp) is provided for users through either a web browser[2] or mobile apps to experience the high-end PC games over the Internet from Shanghai, Shenzhen, and Beijing. The distances between Chengdu and these three cities are about the same, approximately 1800 to 1900 km.

Delays

Delays are distributed among gaming scene rendering, XCODER processing, and network delivery, as shown in Fig. 4. Game rendering usually occupies a few milliseconds. XCODER processing demonstrates various delays due to the different codec implementations. We measure the XCODER processing delay τ_p from the GPU frame buffer to the final compressed stream output for a 1080p gaming video at 30 FPS. For each frame, GPU frame buffer copying takes about 1 ms, followed by the direct GPU encoding, about 5 ms, resulting in $\tau_p^{\text{GPU}} = 6$ ms in total. For CPU implementation, raw frames are transferred from the GPU frame buffer to the CPU memory first, consuming about 10 ms. Software encoding delay depends on how many slices can be encoded in parallel. Our experiments show that 8 slices per frame achieves the best performance at 3.6 ms, resulting in $\tau_p^{\text{CPU}} = 14.6$ ms. For HWA implementation, even though dedicated memory access (DMA) is used, τ_p^{HWA} is about 13 ms in total. Even for the CPU solution, its 14.6 ms is about 11 percent of the 136.7 ms processing delay reported for the OnLive system [13].

The number of slices per frame only impacts the software encoding, but not the GPU and dedicated HWA. Memory transfer latency can be reduced via the light compression inside the GPU. For instance, we can use fast JPEG encoding to achieve 10× compression ratio. Ideally, it needs about 10 percent duration comprared to the existing raw frame exchange scheme. But we have to incorporate a fast JPEG decoder into the HWA and CPU to perform the transcoding rather than the encoding. JPEG encoding or decoding could be less than 1 ms for a well optimized solution. Therefore, we expect $\tau_p^{\text{HWA}} = 6$ ms and $\tau_p^{\text{CPU}} = 7.6$ ms after incorporating the lightweight compression.

Network delay is simulated in an intranet setup to avoid unexpected traffic over the Internet. The intranet is connected using the 1000 Mb/s wired connection. Normally, its delay is less than 1 ms. We use clumsy [14] to control the intranet delay scale. We invited 10 gaming fans to play League of Legends (i.e., one of the most profitable games in China) with different network delay setups, and asked them to provide subjective feedback about the delay configuration, as listed in Table 1. If network delay τ_n is less than 20 ms, no one perceives the difference between playing games over the network or locally. If τ_n is between 20 and 50 ms, it still meets the gaming requirements for the majority of players. Gaming experience is not sufficient when 50 ms < τ_n < 100 ms, and delay is unbearable when τ_n > 100 ms.

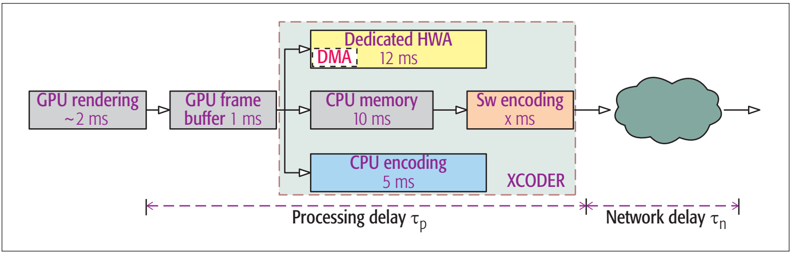


FIGURE 4. Illustration of the delays from rendering to compressed stream output.

Network delay	Users' feedback
< 20	Excellent experience without noticing delay
20–50	3 out of 10 feel some delay
50–100	7 out of 10 feel obvious delay
> 100	Too bad. Gaming is over!

TABLE 1. Network delay τ_n (ms) impact on users' gaming experience.

In addition, we performed the delay measurements between Chengdu (where the TG-cloud is hosted) and Shanghai, Beijing, and Shenzhen, respectively, at a frequency of every half an hour from 9:00 a.m. to 9:00 p.m. The averaged delay between Chengdu and Beijing is about 53.5 ms, while they are 42 ms and 44 ms between Chengdu and Shanghai,and between Chengdu and Shenzhen, respectively. Together with the above gaming field tests, even with the TG-cloud hosted at Chengdu, we can still offer acceptable cloud gaming quality to other cities. The quality could be further improved with more TG-cloud DCs geo-distributed close to the gaming users.

Network Bandwidth Consumption

Network bandwidth consumption mainly depends on the compression methods and gaming content characteristics. We ran the 3D game World of Warcraft (WoW) and performed the encoding using the GPU codec directly. For a 1080p resolution video at 30 FPS, we demonstrated that 10 Mb/s bandwidth is required for an H.264/AVC compatible stream, but 5 Mb/s is sufficient for an HEVC compatible stream to offera high-quality gaming experience.

The above is for a typical gaming scenario. We also measure the "extreme case" where we switch the user view as fast as we can. In this demonstration, bandwidth consumption jumps to 100 Mb/s for the H.264/AVC codec. This is mainly due to the failure of the temporal prediction because of the fast view switch. HEVC only requires about 20 Mb/s for fast view switch because of its superior coding efficiency.

Bandwidth can be further reduced by about 20 to 40 percent bit rate at the same perceptual quality through incorporating the behavior learning statistics.

[2] Google Chrome is recommended currently.

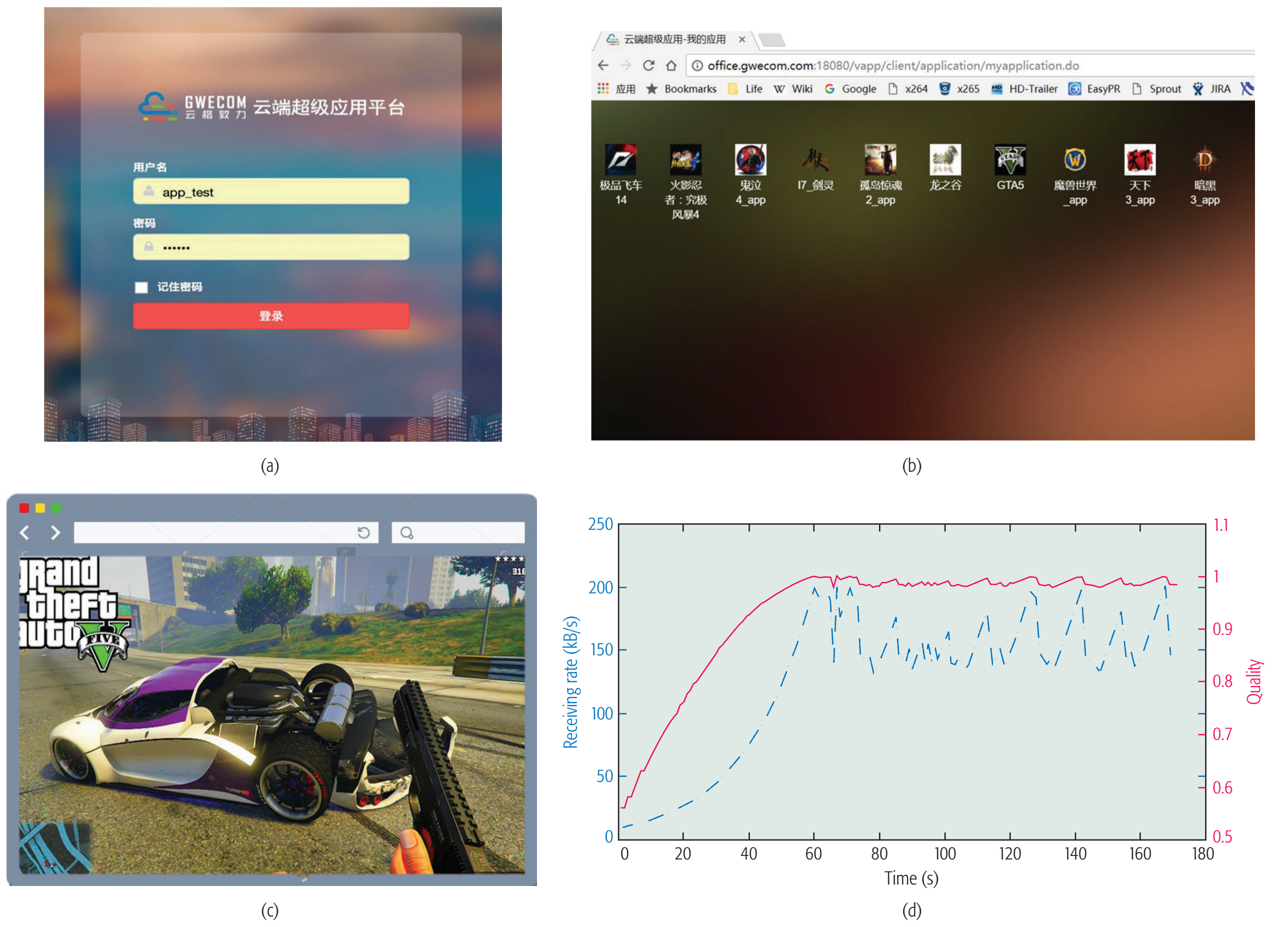


FIGURE 5. Illustration of the TG-cloud example: a) unified web access; b) virtualized game applications; c) instantaneous gaming scene; d) quality-bandwidth adaptation.

Quality-Bandwidth Adaptation

Previously, we developed an analytical model [15] to show the relationship between video stream bit rate R and its perceptual quality, that is,

$$Q(R) = \frac{\left(1 - e^{-\kappa\left(\frac{R}{R_{\max}}\right)^{0.55}}\right)}{(1 - e^{-\kappa})}, \tag{1}$$

with κ as the video content dependent model parameter, ranging from 2 to 6 for typical samples (i.e., from stationary to motion intensive). $R_{\max}$ is the maximum bandwidth allocated to a single user.

In 4G mobile test, we set 200 kB/s as the highest sustainable bandwidth. Server encoding is adapted according to the periodic channel probe from the client. As shown in Fig. 5d, the connection starts from a very low bit rate, and the quality is improved as long as the bit rate is increasing. Once it reaches the ceiling, it fluctuates because of the congestion-induced packet drop. A user's visual system typically tolerates a certain degree of packet drop (e.g., frame drop with slight frame rate variation) without noticeable visual degradation. Hence, the quality fluctuation the user observes is much smaller than the rate variation as captured in the simulation.

GRID vs. TG-SHARE

Since most existing vendors adopt the GRID-based vGPU in offering cloud gaming, we perform a comparison between GRID vGPU using the Telsa K2 and our TG-SHARE using the GTX1080. A gaming scene is encoded using the GPU codec directly. We have evaluated the top 10 games both in China and globally. All games can run on both platforms. TG-SHARE GTX1080 can support six users simultaneously, but only four users on the GRID K2 platform. However, the GRID K2 GPU is about two times more expensive than the GTX 1080. Therefore, the cost of TG-SHARE per user is less than 25 percent of the per user cost of using the GRID K2. Meanwhile, when applying the highest quality settings for each game, gaming FPS is almost two times for TG-SHARE GTX1080 over GRID K2.Besides, webRTC is more convenient for the end user to access the games through their heterogeneous devices.

Concluding Remarks

This work introduces a transparent gaming system, which can be implemented at network edges to facilitate cloud gaming at scale. Compared to the existing cloud gaming technologies, we have developed an innovative GPU sharing scheme, TG-SHARE, to use consumer-level GPU devices without resorting to professional and expensive

GPUs (e.g., GRID compatible cards). In the meantime, gaming scenes are compressed adaptively by leveraging the network dynamics, learned gaming behaviors, and underlying resources. Google webRTC is adopted to encapsulate the compressed streams for real-time interactions and pervasive access from remote heterogeneous devices.

Compared to the existing cloud gaming vendors using the GRID technology, our proposed TG-cloud could save about 75 percent computing expense per user and 20 to 40 percent network expense per user (assuming the network cost is linearly related to the bandwidth consumption).

Compared with the existing cloud gaming vendors using the GRID technology, our proposed TG-cloud could save about 75 percent computing expense per user and 20 percent to 40 percent network expense per user (assuming the network cost is linearly related to the bandwidth consumption).

Acknowledgment

This work was supported in part by National Natural Science Foundation of China (NSFC) projects (Grants 61571215 and 61650101), and in part by the National Science Foundation for Young Scholar of Jiangsu Province, China (Grant BK20140610).

References

[1] "2016 Global Games Market Report: An Overview of Trends and Insights," https://cdn2.hubspot.net/hubfs/700740/Reports/Newzoo Free 2016 Global Games Market Report.pdf, June 2016.
[2] W. Cai *et al.*, "A Survey on Cloud Gaming: Future of Computer Games," *IEEE Access*, vol. 4, Nov. 2016, pp. 7605–20.
[3] S. Shirmohammadi *et al.*, "Introduction to the Special Section on Visual Computing in the Cloud: Cloud Gaming and Virtualization," *IEEE Trans. Circuits and Systems for Video Technologies*, vol. 25, no. 12, Dec. 2015, pp. 1955–58.
[4] J. Ren *et al.*, "Serving at the Edge: A Scalable Iot Architecture Based on Transparent Computing," *IEEE Network*, Aug. 2017, pp. 12–21.
[5] Y. Zhang *et al.*, "A Survey on Emerging Computing Paradigms for Big Data," *Chinese J. Electronics*, vol. 26, no. 1, Jan. 2017, pp. 1–12.
[6] J. Ren *et al.*, "Exploiting Mobile Crowdsourcing for Pervasive Cloud Services: Challenges and Solutions," *IEEE Commun. Mag.*, vol. 53, no. 3, Mar. 2015, pp. 98–105.
[7] R. Shea, D. Fu, and J. Liu, "Cloud Gaming: Understanding the Support from Advanced Virtualization and Hardware," *IEEE Trans. Circuits and Systems for Video Technologies*, vol. 25, no. 12, Dec. 2016, pp. 2026–37.
[8] T. Wiegand *et al.*, "Overview of the H.264/AVC Video Coding Standard," *IEEE Trans. Circuits and Systems for Video Technologies*, vol. 13, no. 7, July 2003, pp. 560–76.
[9] J.-R. Ohm *et al.*, "Comparison of the Coding Efficiency of Video Coding Standards — Including High Efficiency Video Coding (HEVC)," *IEEE Trans. Circuits and Systems for Video Technologies*, vol. 22, no. 12, Dec. 2012, pp. 1669–84.
[10] H. Strasburger, I. Rentschler, and M. Juettner, "Peripheral Vision and Pattern Recognition: A Review," *J. Vision*, vol. 11, no. 13, May 2011, pp. 1–82.
[11] N. Kato *et al.*, "The Deep Learning Vision for heterogeneous Network Traffic Control: Proposal, Challenges, and Future Perspective," *IEEE Wireless Commun.*, vol. 24, no. 3, June 2017, pp. 146–53.
[12] Z. Fadlullah *et al.*, "State-of-the-Art Deep Learning: Evolving Machine Intelligence Toward Tomorrows Intelligent Network Traffic Control Systems," *IEEE Commun. Surveys & Tutorials*, May 2017, pp. 1–1.
[13] R. Shea *et al.*, "Cloud Gaming: Architecture and Performance," *IEEE Network*, July/Aug. 2013, pp. 16–21.
[14] clumsy, http://jagt.github.io/clumsy/index.html, 2015.
[15] Z. Ma *et al.*, "Modeling Rate and Perceptual Quality of Video as Functions of Quantization and Frame Rate and Its Applications," *IEEE Trans. Circuits and Systems for Video Technologies*, vol. 22, no. 5, May 2012, pp. 671–82.

Biographies

YILING XU received her B.S., M.S., and Ph.D. from the University of Electronic Science and Technology of China in 1999, 2001, and 2004, respectively. She is a full researcher at Shanghai Jiaotong University, China. From 2004 to 2013, she was with the Multimedia Communication Research Institute of Samsung Electronics, Korea. Her research interests mainly include architecture design for next generation multimedia systems, cross-layer design, and dynamic adaptation for heterogeneous networks.

QIU SHEN received her B.S. and Ph.D. from the University of Science and Technology of China in 2004 and 2009, respectively. She is now on the faculty of Nanjing University, Jiangsu, China. Her research focuses on video compression, vision modeling, and video tracking.

XIN LI received his B.S. and M.S. from Huazhong University of Science and Technology, Wuhan, China, in 2003 and 2005, respectively, and an M.S. from George Mason University in 2013. He is now the CTO of a startup company that focuses on cloud gaming and gigapixel streaming. From 2012 to 2015, he worked at Samsung Research America. From 2005 to 2008, he worked at the Intel China Software Development Center.

ZHAN MA received his B.S. and M.S. from Huazhong University of Science and Technology, Wuhan, China, in 2004 and 2006, respectively, and his Ph.D. degree from New York University in 2011. He is now on the faculty of Nanjing University. From 2011 to 2014, he was with Samsung Research America and Futurewei. His research focuses on video compression, gigapixel streaming, and multi-spectral signal processing.