Interactive Screen Video Streaming-Based Pervasive Mobile Workstyle

Zhan Ma, *Member, IEEE*, Tao Yue, Xun Cao, *Member, IEEE*, Yiling Xu, Xin Li, and Yongjin Wang

***Abstract*—In this paper, we develop an interactive screen video streaming-based system to enable the ubiquitous mobile workstyle, which is referred to as personal computer to pervasive computing (PC2PC). The desktop screens of virtualized systems are compressed in the PC2PC servers and delivered to remote end users for stream decoding, rendering, and interactions. We have implemented a system from the scratch, where the emerging screen content coding extension of high-efficiency video coding is implemented to compress and stream the desktop screens of the virtualized system in real time. Three core asset channels, system, display, and inputs, are defined to enable systematic end-to-end communication. Compared with Red Hat SPICE virtual desktop infrastructure scheme, the proposed PC2PC could save network bandwidth consumption by a factor of 2, 7, and 4, respectively, in terms of typical video streaming, web browsing, and stationary office applications at the same visual quality. Meanwhile, we have also measured the delays of the system and presented preliminary results on the user experience aspect. A simple network estimation is applied to optimize the quality bandwidth adaptation for both single user and multiuser scenarios to consider the network dynamics.**

***Index Terms*—High-efficiency video coding, pervasive computing, screen content coding, virtual desktop infrastructure.**

I. Introduction

MOBILE workstyle is an emerging office fashion that allows employees to manage and operate their daily tasks remotely. Such mobile workstyle is supported by the *virtual desktop infrastructure* (VDI) where routine desktop operating systems (OSs) are hosted at data centers (i.e., private or public cloud) to provide the ubiquitous computing and storage. It enables people to strike a better work-life balance and work from anywhere, unrestricted by location, time or device [1], [2]. Meanwhile, it also provides better security by concealing sensitive and confidential information processed and stored in data centers.

Industry leaders in the area of information technology (IT) actively embrace the mobile workstyles and have developed several well-known platforms to enable such service, for instance, VMware Horizon [3], Citrix XenDesktop [4], Microsoft Remote Desktop Protocol (RDP) [5], Red Hat Simple Protocol for Independent Computing Environments (SPICE) [6], etc. These popular VDI technologies follow the similar procedure: an instantaneous desktop screen is first analyzed to identify different regions, such as text, icon, graphics, images, etc, and each region will be processed individually using compression methods for network delivery. They require complex protocols to maintain the communication between the client and remote server. Because of the noticeable involvements from the client for desktop rendering, it is also referred to as the "client rendering" solution.

Due to the advances in communication (such as 4G/LTE, emerging 5G) and mobile computing capability, we envision the increasing market demand of the mobile workstyle through the VDI technologies. However, most existing technical solutions are not well suited due to high complexity of "client rendering" protocols. In this work, we propose a complete "server rendering" solution where the server renders and compresses the instantaneous desktop screen, and the client only decodes the compressed screen stream and sends back mouse and keyboard commands for user interaction. This is motivated by the fact that *instantaneous desktop screen rendering is actually a video playback application refreshed at 60 frames per second (FPS)*[1] for a typical display.

Note that offering the pervasive mobile workstyle requires substantial effort in various aspects, such as OS virtualization, cloud load balancing, VDI, etc. In this work we focus on VDI that is applied to enable the interaction between hosted OSs and remote clients. The overall PC2PC system is illustrated in Fig. 1. A real-time HEVC based SCC compression engine encodes instantaneous desktop screens of a virtualized OS at the server side. This is referred to as the **Display Channel** to deliver the compressed streams. To enable the systematic end-to-end communication, we have also defined another two asset channels, i.e., the **System Channel** to convey the messages so as to initiate, maintain/monitor and destroy the communication

Manuscript received January 14, 2017; revised June 21, 2017 and August 3, 2017; accepted August 3, 2017. Date of publication August 9, 2017; date of current version September 15, 2017. This work was supported in part by the National Natural Science Foundation of China (NSFC) Projects under Grant 61371166 and Grant 61422107, in part by the NSFC Projects under Grant 61571215, Grant 61650101, Grant 61322112, and Grant 61531166004, in part by the National Science Foundation for Young Scholar of Jiangsu Province, China, under Grant BK20140610, and in part by the Natural Science Foundation of Jiangsu Province under Grant BE2016186. The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Shiwen Mao. *(Corresponding author: Yiling Xu.)*

Z. Ma, T. Yue, and X. Cao are with Nanjing University, Nanjing 210008, China (e-mail: mazhan@nju.edu.cn; yuetao@nju.edu.cn; caoxun@nju.edu.cn).

Y. Xu is with Shanghai Jiaotong University, Shanghai 200240, China (e-mail: yl.xu@sjtu.edu.cn).

X. Li is with Yun Ge Zhi Li, Inc., Richardson, TX 75082 USA (e-mail: lixin@gwecom.com).

Y. Wang is with the Nanjing University of Posts and Telecommunications, Nanjing 210028, China (e-mail: wangyj@njupt.edu.cn).

Color versions of one or more of the figrues in this paper are available online at http//ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2017.2737944

[1]Different LCD/LED displays may have different refresh rate in practice.

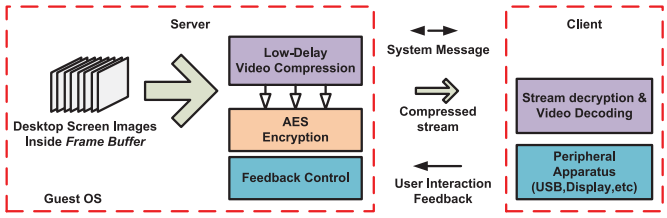


Fig. 1. Illustration of our proposed PC2PC architecture and implementation.

channel and the **Input (feedback) Channel** to collect the user interaction commands (e.g., keyboard, mouse, joysticks, etc) from the client. TCP is used to manage the stream delivery between the server and the client.

This work mainly focuses on the interactive screen video system design and analysis, which is a typical video application over the network. To the best of our knowledge, this is the first work that proposes to use (and actually has implemented) recently approved screen content coding standard to compress instantaneous screens and stream to remote users over the Internet. Different from the existing client rendering solutions, this is a complete server rendering scheme with managed computing at the server farm exclusively and an ultra-light-weight client just for user interaction. Our proposal significantly reduces the protocol complexity to enable the VDI service in reality.

Meanwhile, VDI has become a typical interactive video streaming optimization to maximize the visual quality with the constraints of both network bandwidth and delay. Conventionally, video (or media) researchers mainly focus on the trade-off between network bandwidth and quality (i.e., live streaming such as Netflix or YouTube). Delay is another crucial factor since VDI offers the interaction between the user and remote server. Thus, how to jointly optimize quality with respect to the network bandwidth and delay is an interesting and inevitable challenge.

Towards this end, we then implement such interactive video streaming based VDI system from the scratch and perform the preliminary study on its bandwidth consumption, delay measurement as well as the quality-bandwidth optimization. With these studies, we would like to encourage more researchers to investigate this interactive video streaming problem and develop advanced algorithms to further enhance current implementation [7]. On the other hand, we have already shown quite an impressive improvement compared with the conventional SPICE platform, with the bandwidth consumption reduction by a factor of 2, 7 and 4 for typical video streaming, web browsing and stationary office applications at the same visual quality. This promises the brilliant future of such system from the theoretical analysis to the practical application. Thus, we believe this work holds significant impacts to media society and sheds some insights as summarized in the paper.

The remainder of this paper is organized as follows: Section II reviews the functions in SPICE protocol. Section III introduces the PC2PC architecture followed by its implementation detailed in Section IV. A comparative study is performed in Section V to demonstrate the performance of proposed system and finally we will conclude this work and discuss a few future directions in Section VI.

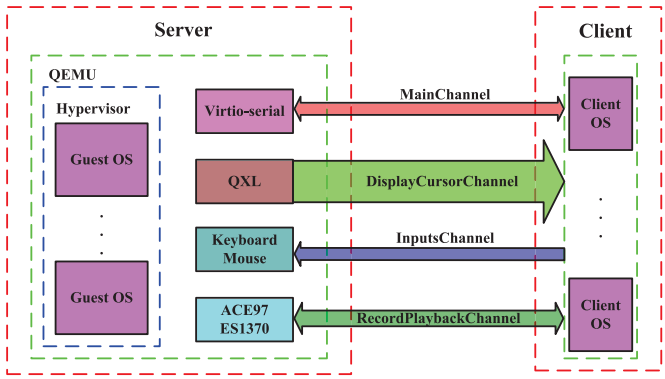


Fig. 2. Illustration of open-source SPICE architecture.

II. Related Work: A Glance of Red Hat SPICE Protocol

Pioneers have devoted tremendous efforts to manage the computer for work remotely. It starts from the basic shell command from the legacy UNIX time. Then it comes to the windows time using remote display, such as the famous RDP developed by Microsoft, SPICE supported by Red Hat, etc. SPICE is an open standard while RDP is a proprietary solution owned by Microsoft. We use the SPICE function as the example. RDP and other protocols [3], [4] share similar architecture.

SPICE is built for virtual OS to allow the end user to connect the server from anywhere, at anytime, through any device. There are multiple data channels defined to carry the command messages and information from the server desktop to the client side. Fig. 2 shows the data channels specified in SPICE protocol. As shown, guest OS is hosted using QEMU [8] hypervisor, and communication between the server and the client is bridged by multiple channels. Each channel is dedicated to a specific type of data. For example, main channel is for system control and configuration; display channel is for graphic commands, images and video streams; inputs channel is for keyboard and mouse feedback; record and playback channel is reserved for audio processing and communication, and so on. SPICE client requires extensive interaction with the hosted OS and SPICE protocol stack to interpret messages and render the server screen over the network properly.

Display channel would consume a great amount of transmission bandwidth because of the large amount of data involved in the delivery of rich screen content over the network. Basically, the server sends messages with primitive graphical commands and data to the client to draw/update/reset the area for rendering. OS dependent items, such as icon, window, glyphs, etc, are rendered at the client side using various message commands. For raw image, SPICE can apply the QUIC or LZ [9] methods for compression directly. Alternatively, SPICE introduces the palette and pixel map (i.e., index map) representation of the raw image, where LZ algorithm (and its variants[2]) are adopted to compress the palette and pixel index map [10]. Besides, it has a "copy mode" in which images are cached at the client side

[2]LZ with dictionary is used to improve the compression performance.

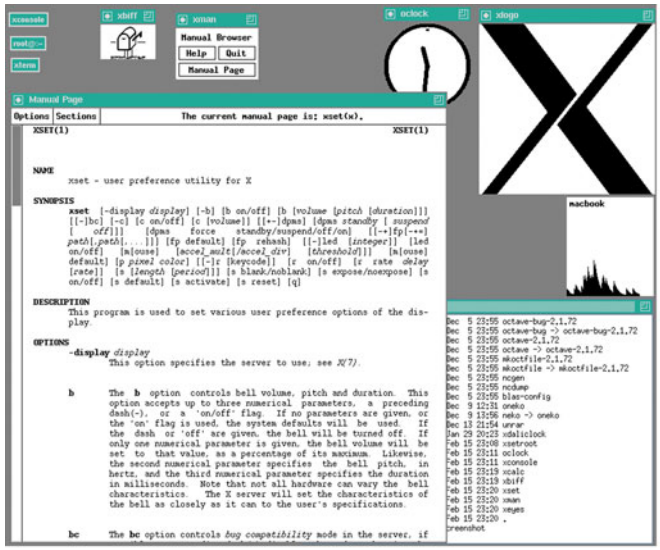
so that they can be directly retrieved for re-rendering at a later time under the assumption that images may be presented on the screen for an extended period of time. Moreover, a video area could be lossy compressed using a specific video codec (such as motion JPEG, MPEG, etc). Users demand the smooth (without flickering, motion jerkiness, freezing frame, etc) video playback for a pleasant quality of experience (QoE) at clients. Often times, buffers are utilized, but sometimes frames have to be dropped due to the severe network congestion. Frame-dropping may introduce the QoE degradation due to the motion discontinuity. How many frames can be dropped in a second without any loss of perception of quality is content dependent [11]. For motion intensive videos, such as sports or gaming [12], higher frame rate (30 frames per second [FPS] or more) is preferred while lower frame rate (15 FPS or less) can still offer sufficient quality for stationary content like conferencing. Alternatively, frames can be heavily compressed to trade-off spatial quality for temporal motion smoothness without dropping frames [13].

III. Architecture of Personal Computer to Pervasive Computing (PC2PC)

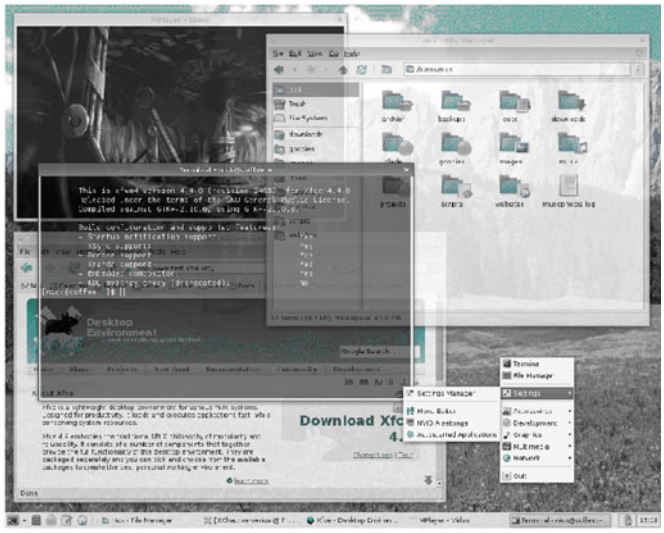
Most existing VDI solutions are based on the client rendering framework where complicated protocols are required to maintain multiple data communication channels (see Fig. 2) between the server and client. It has noticeable system complexity when applied in practice. Meanwhile, another crucial issue that limits the massive adoption of existing VDI solutions over the Internet is the large network bandwidth consumption. Fig. 3 illustrates the typical desktop screen snapshots. As we can see, SPICE or similar methods could well present the desktop window items at old times because of its simple structural layout, color distribution, etc. For instance, text rendering is just binary black and white as shown in Fig. 3(a), which is much easier to signal with limited bandwidth requirement using the glyphs commands. However, advanced sub-pixel rendering techniques [14] are developed nowadays to enhance the text quality and to improve the user experience [see Fig. 3(b)]. With these advanced technologies, desktop elements, such as icon, text, glyph, etc are built up with more complex structural layout and color mixtures, resulting in significant bandwidth requirement if we want to display the high-fidelity desktop screens at the client over the network.

Traditional remote desktop control in the computer architecture area is actually an *interactive video streaming* (i.e., encoding, delivery, decoding, and interaction) problem in the media society. Ideas that have been developed so far for efficient video compression and streaming could be borrowed extensively to enhance the system.

We build the PC2PC framework from the scratch with three core asset channels to fulfill the required functions. System channel is used to deliver messages for communication initialization, maintenance/monitoring and destruction; SCC is used for screen compression in display channel while input channel offers the capability to manage the end user interaction. Other channels could be extended in future to support additional functionalities, such as USB, network sharing folder, printer, etc.



(a)



(b)

Fig. 3. Desktop representation example in (a) old times and (b) modern times.

As shown in Fig. 1, PC2PC is much simpler than the legacy SPICE/RDP solution shown in Fig. 2. Note that with the proposed PC2PC framework, we provide the complete "server rendering" solution in comparison to the current client rendering methods.

Existing VDI technologies first analyze the desktop screens into various regions with different content, then apply multiple schemes (with or without compression) to represent these regions in display channel. They require that messages are explicitly defined to include all possibilities (such as text, image, glyph, icon, etc) for a specific region by its attributes (i.e., position, area, characteristics, etc). Together with other data channels, a great amount of messages and their associated operational protocols (e.g., synchronization) have to be carefully followed to ensure the end-to-end communication [6]. Messages are encapsulated at the server (in a pre-defined manner) and interpreted at the client. For some of them, particularly those

messages associated with the display channel, client OS related processes are often triggered to facilitate the rendering, such as to draw/update/reset screen regions.

From the client rendering to server rendering, we significantly simplify the overall system implementation without requiring the complicated protocols (particularly for the display channel). Instantaneous screen frames are encoded using a single SCC encoder without complex region segmentation and processing. Client could be an ultra-light-weight device with just video stream decoding and user interaction capabilities.

IV. Implementation Details of Proposed PC2PC System

This section unfolds the implementation details of the proposed PC2PC system.

A. Screen Video Compression

As aforementioned, a computer screen is another typical video frame. Rather than the traditional camera captured natural video, it is with mixed resolutions of text, icon, glyphs, graphics, image, video, etc. It plays at a fixed frame rate at 60 frames per second aligned with the display refresh rate. Legacy coding standards, for instance, H.264/AVC [15], HEVC [16], VP9 [17], AVS [18], etc., are primarily developed for the continuous-tone camera captured video. The computer screen mixes discontinuous-tone content such as text, icon, graphics, and continuous-tone content such as video sequences and images with natural content. Continuous-tone and discontinuous-tone content have quite different statistical distributions [19], [20]. For instance, pixels in continuous-tone content evolve smaller intensity changes among local neighbors while neighboring pixel intensity could vary unexpectedly for discontinuous-tone samples. Furthermore, discontinuous-tone content may contain less distinct colors than the rich color distribution in continuous-tone content. On the other hand, local samples of continuous-tone content typically present more complex textures and orientations compared with the discontinuous-tone content. For instance, HEVC employs up-to 35 modes for spatial intra prediction to encode the natural content [16]. It might be good enough to have few angular modes (such as horizontal, vertical, etc) since screen content typically contains well structured local orientations [10].

It motivates the development of new tools to leverage the characteristics of screen content which are not exploited in natural camera-captured videos to compress the screen content more efficiently in both the academia and the industry [19]–[22]. Furthermore, the international standard committee decided to launch the screen content extension (SCC) development based on the recent HEVC framework after evaluating multiple Call-for-Proposal responses [23] in April 2014. After several meeting cycles, key tools that provide substantial coding efficiency improvement and reasonable complexity trade-off, have been adopted into the standard specification for the final approval of the SCC recommendation in March 2016. These tools includes intra block copy (intraBC) [24]–[26], color table/palette coding [27]–[30], adaptive motion vector resolution [31], color transform [32], etc. Ideally, SCC profile of the HEVC would provide the state-of-the-art compression performance for screen applications.

TABLE I
Basic Tools in Our Real-Time SCC Encoder

Tools	Description
Input Source	8-bit YUV 420
Prediction Structure	IPPP
Coding Unit	64×64 to 8×8
Prediction Unit	$2N \times 2N$, $N \times 2N$, $2N \times N$
Transforms	32×32 to 4×4 (no recursive decision)
Filtering	deblocking, sample adaptive offset
Modes	spatial intra, inter, intraBC, palette
Motion Vector	integer MV resolution (no interpolation)
Parallelism	Multiple slices

To analyze and understand the impact of applying video compression for VDI solutions, we implement the emerging SCC into our proposed PC2PC framework to provide the ubiquitous computing environment. The encoder is configured with a set of simplified parameters as shown in Table I in order to provide the required smoothness and low-delay for real-time application.

Even though screens are typically rendered in the 8-bit RGB format, we use the YUV420 as the input video source in our study to demonstrate our ideas. Color format conversion (i.e., RGB-to-YUV420 in encoder and YUV420-to-RGB in decoder) is performed independently using the libyuv [33]. This is mainly because YUV420 content requires less computational resource and bit rate as compared with the corresponding RGB samples [34]. In the meantime, in order to quickly demonstrate our ideas, we refer to the encoder implementation optimization in FFmpeg [35] where most of them focus on the YUV420 color space. RGB content has its advantages to maintain the signal fidelity, particularly for areas with sharp edges (e.g., texts) [10]. In the next step, we plan to extend the current implementation to support the RGB color space.

Multiple reference frames are used in both H.264/AVC [15] and HEVC [36] to improve the coding efficiency by searching the best candidate multiple times in different reference frames. In our SCC encoder, we constrain the encoder to only use the nearest decoded reference picture.

Low-delay B picture is adopted in HEVC by allocating the same references in both forward and backward reference lists. Because of the recursive block matching, it provides 6-8% [36] compression gain compared with the conventional low-delay P structure. Concerning with the almost 3x complexity increase, we use the legacy P picture rather B picture for low-delay prediction.

Recursive coding unit (CU) [37] is implemented from 64×64 to 8×8 given that it contributes the significant performance improvement from the flat macroblock structure in H.264/AVC [15]. Prediction unit (PU) is coupled with the CU for intra mode, while only symmetric partitions is enabled in inter mode, i.e., $2N \times 2N$ CU comes with the $2N \times 2N$, $2N \times N$ and $N \times 2N$ PUs. There is no PU size less than 8×8. Furthermore, transform unit (TU) is also attached to the CU with only two-level decision, i.e., a $2N \times 2N$ CU will try $2N \times$

2N and N × N. Recursive quad-tree transform is not supported in the current implementation.

In addition to the conventional spatial intra and temporal inter modes, intraBC and palette modes are included to efficiently process the screen contents. Note that we use a small local search window [10] (i.e., 3 × 3) for intraBC mode rather than the full frame search in SCC reference model. Only integer motion vector (MV) resolution is enabled right now without the requirement of the interpolation filter. Both deblocking and sample adaptive offset (SAO) are equipped to improve the visual quality.

Slice mechanism is used to provide the parallel processing. Note that in order to quickly demonstrate our baseline prototype, encoder parameters are almost all fixed (such as the CU sizes) except the slice number. We allow the flexibility of the slice number to evaluate the delay impact in Section V-A3.

These algorithm simplifications and appropriate platform optimization (such as SSE and MMX) enable real-time encoding of the SCC content on general purpose CPUs.

B. A Simple Network Bandwidth Estimation

We often face the network dynamics because of the congestion, channel fading, etc. The key issue to maintain the quality of the service is adapting the sending rate based on the real-time network status probing.

Google Congestion Control (GCC) algorithm offers great performance and is adopted in WebRTC framework for real-time communication. GCC employs two controllers: a sender-side controller, which computes the target sending bitrate A_s, and a receiver-side controller, which computes the rate A_r that is returned to the sender. The sender will adjust the sending bitrate to the $\min(A_s, A_r)$. 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.

In our baseline system, we provide a simplified version of the GCC where only the client side bandwidth estimation is utilized. Such network status is conveyed in system asset channel to adapt the server sending rate. Our tests have shown that video consumes over 95% of the total network bandwidth. Conservatively, we scale the estimated sending rate by 0.9 as the screen compression bit rate. Network dynamics are inferred by the estimated sending rate.

As an aggressive strategy, GCC will consume any residual bandwidth. So the "best" quality of all users can not be guaranteed. Allocating each user equal bandwidth is the easiest but obviously not the best solution because different services have different requirements for network resources. Some efforts have been made to provide a good solution of bandwidth allocation for competitive users, such as the game theoretic framework [38] and dynamic bandwidth allocation [39].

For VDI service, the server is aware of the screen content characteristics of different users. Hence, the server can manage network bandwidth according to the individual application service. A simple solution is to allocate the network resources to get the best video quality (such as Peak Signal-to-Noise Ratio PSNR) balance among all the users. Intuitively, more bandwidth should be given to the user who is watching videos (e.g., high frame rate and high bit rate case) than the user who is just editing a slides (e.g., lower frame rate and lower bit rate case).

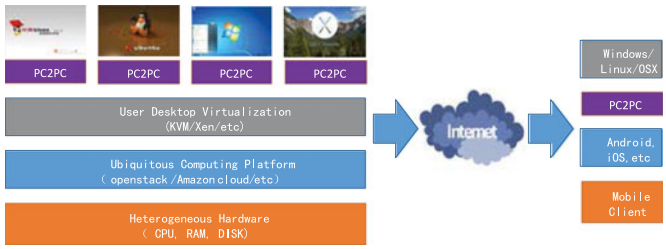


Fig. 4. Illustration of the PC2PC-based private cloud service.

C. Client Implementation

At the client side, we choose the Firefly RK3288 [40] development board to implement our PC2PC client algorithm. All functions are realized in pure software fashion on its Quad-core ARM platform. Since the stream is compliant with the emerging SCC standard and there is no ASIC chip available in the market yet, it is preferable to use the optimized software decoder. Basically, PC2PC client decodes the received stream and renders it on the display. Meanwhile, it also captures the user interaction commands, such as the keyboard and mouse messages, and sends back to the server. Since the decoded screen image is YUV420 format, color conversion is performed to translate it to the RGB space before rendering it on display. Such color conversion is also conducted on the ARM core, which requires substantial computing power, particular for videos with higher spatial resolution, i.e., 1080p or even 4K content. A potential improvement is shifting the color conversion to the GPU that will offload the ARM computing resource massively. Since Firefly RK3288 offers similar computing power as the recent tablets and SmartPhones, we can migrate our implementation on those platforms as well.

V. Experimental Study and Discussion

Two major tests are carried out to demonstrate the performance of our proposed PC2PC system. One performs the actual field tests for practical virtual desktop applications using our PC2PC platform as well as the default SPICE system. The other test discusses the quality-bandwidth adaptation for our PC2PC system.

A. Office-in-Pocket: PC2PC Versus SPICE

We perform the experiments by collecting the actual network bandwidth consumption in real life applications for commercial Red Hat SPICE system and our proposed PC2PC platform. Towards this goal, we build two small private clouds shown in Fig. 4. These two clouds are hosted at N-city and S-city with 200 miles distance, respectively. Each one uses 10 high performance servers. Each server is equipped with two 20-core Xeon

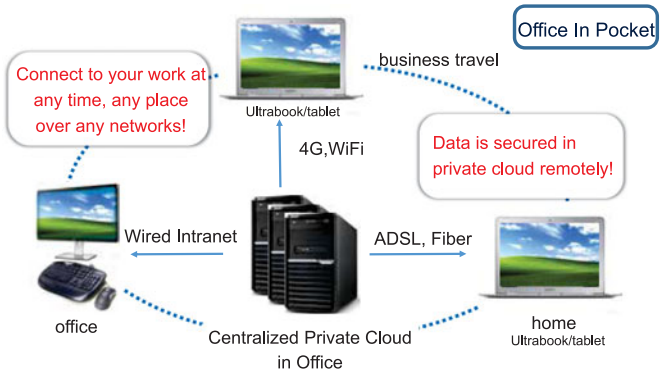


Fig. 5. Illustration of the mobile workstyle – "office in pocket": our PC2PC server is deployed on centralized servers and PC2PC client could be implemented on any mobile devices, such as dedicated mobile platform (i.e., Firefly RK3288 as an example), tablet, SmartPhone, ultrabook, etc.

CPUs, 128 Gigabytes RAM and 2 Tera-bytes hard disks. Servers are interconnected using the 10G Ethernet.

With the above private cloud setup, we can enable the pervasive computing to fulfill the actual "office in pocket" shown in Fig. 5. PC2PC framework provides the ubiquitous way for users to connect their office environment at any time, any place over any network. For instance, users can connect to the office server using wired Ethernet when they are present in office; users can also log into the office desktop using home network such as ADSL or Fiber; users can even continue their office work without interruption during the business travel through 4G or Wi-Fi networks. Data between the server and the client in PC2PC system can be optionally encrypted to provide enhanced secure computing and communication.

1) Wireless Bandwidth Measurement: Before jumping into the actual performance comparison between Red Hat SPICE and our PC2PC platforms, we have implemented a daemon to monitor the wireless network bandwidth (i.e., Wi-Fi, China Mobile 4G and 3G). We measure the Wi-Fi, 4G and 3G bandwidth consumptions using a PC2PC client at N-city to the N-city PC2PC server farm; while only measure 4G and 3G bandwidth consumptions at N-city to the S-city servers. We measure the network bandwidth every 30 minutes for five weekdays. Network measurement is performed everyday at a frequency of every half an hour from 11 AM to 11 PM, i.e., 11:00 AM, 11:30 AM, 12:00 AM, ..., 10:30 PM, 11:00 PM. To better present the data, we organize the measurement data in terms of the down-link bandwidth BDL, up-link bandwidth BUL and round-trip delay τ in Table II, each of which includes the corresponding mean value, standard deviation, maximal and minimal.

We observe that China Mobile 4G offers impressive wireless connections for both intra and inter city scenarios, with better performance in comparison to the Wi-Fi using the minimum bandwidth as the measurement criterion. Even for inter-city 4G network, it provides 500 KB/s down-link and 130 KB/s up-link bandwidth, respectively, which is sufficient to maintain the stable connection for typical applications using PC2PC platform (cp. Section V-A.2). However, the round trip delay τ of inter-city 4G is significantly worse, about 4x larger for averaged case and 6x larger for maximum delay criterion. This would heavily degrade the user experience, particularly for those delay sensitive applications such as gaming. For those stationary applications, such as word editing, email drafting, etc, higher delay (and lower frame rate) could be tolerated where we can try to drop some frames for better subjective quality spatially [11], [13]. Compared with the Wi-Fi and 4G networks, China Mobile 3G could only enable very limited applications with low bandwidth requirement upon its network conditions.

TABLE II
WIRELESS STATISTICS (DOWN-LINK BANDWIDTH BDL, UP-LINK BANDWIDTH BUL AND ROUND-TRIP DELAY τ)

		Ave.	Std.	max	min
intra-city measurement					
Wi-Fi	BDL (MB/s)	1.1	0.3	1.7	0.3
	BUL (MB/s)	0.6	0.18	0.88	0.14
	τ (ms)	57	15	92	38
4G	BDL (MB/s)	3.5	2.0	5.8	2.4
	BUL (MB/s)	0.6	0.3	1.1	0.13
	τ (ms)	41	15	73	19
3G	BDL (MB/s)	0.18	0.02	0.20	0.14
	BUL (KB/s)	10	6.3	35	4.7
	τ (ms)	125	36	272	85
inter-city measurement					
4G	BDL (MB/s)	3.3	1.8	5.7	0.5
	BUL (MB/s)	0.48	0.29	1.1	0.13
	τ (ms)	154	120	401	45
3G	BDL (MB/s)	0.18	0.02	0.20	0.14
	BUL (KB/s)	10	7.3	36	3.5
	τ (ms)	310	165	796	110

Note that the network delay could be partially resolved by deploying the PC2PC server farm inside the high rank data centers, instead of using current family-type ISP. Usually, advanced data centers are geographically distributed. User's data (e.g., guest OS) can be cached and synchronized among several places. The one producing the least network delay is chosen to perform the work [41].

2) Network Bandwidth Impact: In this section, we mainly compare the (downlink) display channel data bandwidth given that complete different methods for screen display delivery are used in Red Hat SPICE and the proposed PC2PC. System and input channels are implemented in a similar way to the SPICE counterparts (i.e., main and inputs channels respectively) in our PC2PC system. We would expect similar network behavior for both methods. Experiments are carried out to ensure the visually lossless quality (i.e., with about 55 dB Peak Signal-to-Noise Ratio PSNR) for both SPICE and our PC2PC methods.

We have collected the data for three typical scenarios when the users are using Microsoft Windows 7 environments.

1) "office" represents the scenario that a user performs typical daily assignments, such as word editing, slides show, email drafting, etc.
2) "web" is for typical web surfing where users scroll the web pages quickly to track and locate interesting topics.
3) "video" is for the on-line video streaming service (such as YouTube) where users also fast scrolls the web browser to locate a video (at the beginning).

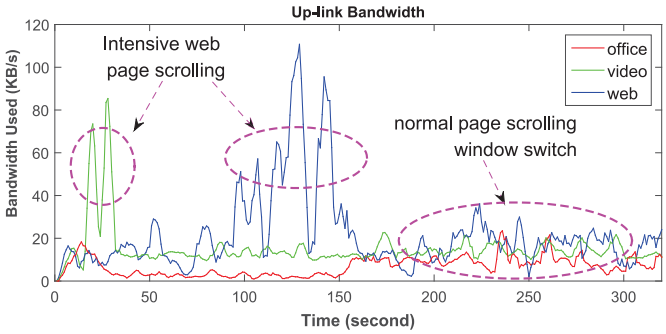


Fig. 6. Up-link bandwidth trace for SPICE and PC2PC system.

It is often very stationary with few keyboard and mouse hits and very slow page scrolling for "office" application, while "web" and "video" applications would have an intensive interaction at the very beginning to fast locate the specific content. PC2PC and SPICE exhibit similar uplink behavior according to the experiments for system and input channels. Fig. 6 depicts the up-link bandwidth for the three use cases. Significant lower bandwidth is required for the stationary "office" application, while the peak impulses are observed for both "web" and "video" scenarios, corresponding to the relative intensive user interaction period for topic retrieval.

Based on the wireless bandwidth measurements shown in Table II, Wi-Fi and 4G networks are feasible to offer sufficient room to support the up-link stream. Even for the minimal bandwidth criterion, Wi-Fi and 4G offers 140 KB/s and 130 KB/s respectively, which is larger than the peak bandwidth requirement of our PC2PC and SPICE up-link shown in Fig. 6. Such peak bandwidth requirement happens when user switches windows or scrolls the pages intensively.

However, 3G up-link can not sustain the service with consistent quality. On the other hand, round-trip delay is also the practical obstacle for service enabling. For instance, it has about 4-frame delay (for a 30 Hz video) on average for intra-city 3G connection, but about 10-frame delay for inter-city 3G network (which makes it impossible to provide the acceptable QoE).

Fig. 7 shows the down-link traces of three application scenarios for both SPICE and PC2PC systems. Note that display data channel in SPICE platform requires a tremendous amount of bandwidth for visual lossless user experience. Particularly the instantaneous bandwidth jumps to about 9.8 MB/s when the screen content has changed significantly due to the intensive user interaction, such as fast content scrolling in "web" application around 125 second shown in Fig. 7(b). As shown in Table III, on average, PC2PC system only requires 25%, 13% and 59% network bandwidth of SPICE protocol for "office", "web" and "video" use cases with the same picture quality, respectively. We also give the maximum bandwidth requirement for both systems, where our PC2PC system consumes 56%, 8% and 23% bandwidth of default SPICE for those scenarios, respectively.

SCC video encoder is deployed to compress the instantaneous desktop screen where the first picture is placed using the intra frame (I-frame) followed by the all forward predictive frames (P-frame). I-frame typically requires more than 5x bits compared with the P-frame. Thus, there is a bit rate jump for the first frame. If the following pictures share the similar content, bit rate consumption will be reduced massively due to the efficient temporal prediction [see 0–10 s of Fig. 7(b) and 7(c)]. But, if the screen refreshes intensively (i.e., window switch, page scrolling) with completely different content, bit rate consumption will jump noticeably (see 0–10 and 150–275 s of Fig. 7(a) where we switch the window very often!). In such circumstance, intra prediction is preferred, rather than more efficient temporal inter prediction. Even if we fast scroll the pages for "web" and "video" applications, since there are still stationary regions (i.e., homogeneous background block, similar texts, etc), temporal prediction still works very well with much less bandwidth requirement compared with the SPICE system (cp. Table III).

Note that SPICE is not feasible to enable the VDI service over the mobile networks because of its significant bandwidth consumption as illustrated in Tables II and III. But both Wi-Fi and 4G networks could offer sufficient bandwidth for our PC2PC system.

3) Delay Impact: There are various delays in the overall system. To fully understand this problem, we plot the processing pipeline in Fig. 8. Delays are categorized into two parts: the server processing delay τ_0 and the network transmission delay τ_1.

Since we only implement the slice threading in our baseline SCC encoder for parallel processing, we evaluate the overall server processing delay using different numbers of slices for 1080p screen video. Both stationary (office and web surfing) and motion intensive (action movie or gaming) screen scenarios are considered with collected data shown in Table IV. When we increase the slice number, we see the server processing delay is decreased noticeably, for instance, 8 slices per frame requires only 50% processing time compared with the 1 slice per frame. Note that the there isn't clear benefits when increasing the slice number from 8 to 16. This might be the reason that 16-thread parallelism requires more system overhead than 8-thread case in our current implementation. Hence, we constrain our SCC encoder using 8 slices per frame.

Network delay is simulated in an intranet setup to avoid unexpected traffic over the Internet. Intranet is connected using the 1000 Mbps wired connection. Normally, its delay is less than 1 ms. We use clumsy [42] to control the intranet delay scale. We have invited 10 gaming fans to play the League of Legends (i.e., one of the most profitable games in China) with different network delay setup, and ask them to provide the subjective feedback about the delay configuration, as listed in Table V. If network delay τ_1 is less than 20 ms, no one perceives the difference between playing games over the network or locally. If τ_1 is between 20 ms to 50 ms, it still meets the gaming requirements for majority players. Gaming experience is not sufficient good when 50 ms $< \tau_1 <$ 100 ms, and delay is unbearable when $\tau_1 >$ 100 ms.

B. Quality-Bandwidth Adaptation

1) Quality-Bandwidth Optimization for Single User Case: Previously, we have developed an analytical model [43] to express the relationship between video stream bit rate and its

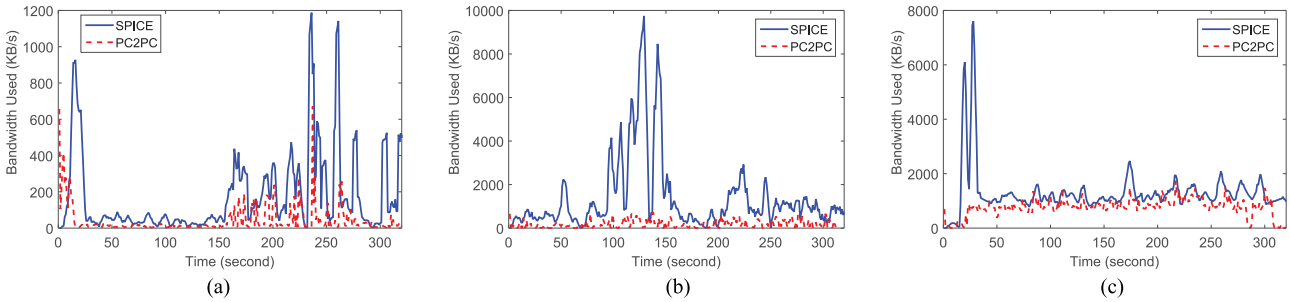


Fig. 7. Down-link bandwidth comparison for SPICE and PC2PC system: (a) office; (b) web; (c) video.

TABLE III
STATISTICS OF NETWORK BANDWIDTH CONSUMED BY THE DEFAULT SPICE AND PC2PC

scenario (KB/s)		ave.	std.	max.	min.
office	SPICE	178	228	1188	0
	PC2PC	44	80	671	3.3
web	SPICE	1401	1743	9750	0
	PC2PC	182	181	717	2.5
video	SPICE	1324	880	7580	0
	PC2PC	770	313	1740	2.8

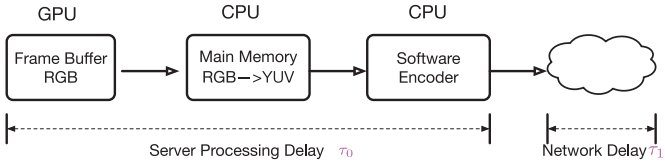


Fig. 8. Potential delays in PC2PC system pipleline.

TABLE IV
SERVER PROCESSING DELAY MEASUREMENT

slice#	stationary (ms)	motion intensive (ms)
1	5.1	8.8
2	3.6	5.5
4	2.9	4.4
8	2.4	3.6
16	2.7	3.7

TABLE V
NETWORK DELAY τ_1 AND OVERALL DELAY $\tau = \tau_0 + \tau_1$ (MS) IMPACT ON USER GAMING EXPERIENCE

network	overall	user feedback
<20	<25	excellent experience without noticing delay
20–50	25–55	3 out of 10 feel some certain delay
50–100	55–105	7 out of 10 feel the obvious delay
>100	>105	Too bad. Gaming is over!

perceptual quality, i.e.,

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

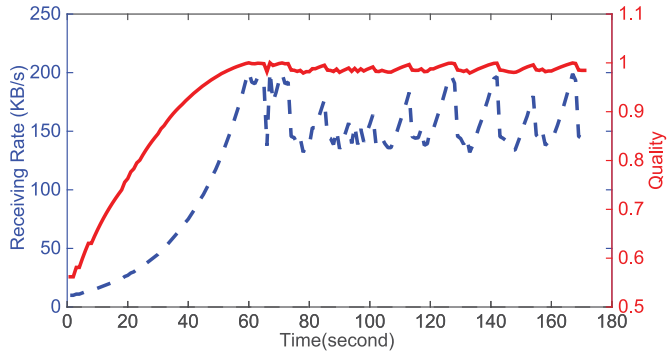


Fig. 9. Quality-rate adaptation using GCC: quality improves as the rate increases; quality fluctuates as the rate varies due to the packet drop but with smaller variation.

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 the 4G test case, we mark 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. 9, the connection starts from the very low bit rate, and the quality is improved as long as the bit rate is increasing. Once it reaches the ceiling, it will fluctuate because of the congestion-induced packet drop. User's visual system typically tolerates a certain degree of packet drop (such as frame drop with slight frame rate variation) without noticeable visual degradation [11]. Hence, we can observe the quality fluctuation is much smaller than the rate variation as captured in the simulation.

2) Quality-Bandwidth Optimization for Multiuser Case: Although GCC can provide good network utilization for a single user, it has obvious drawbacks for typical multi-user applications. Here we would have an upper bound of the total network bandwidth shared by all users. As discussed in previous sections, we could have two schemes to do the bandwidth allocation shown in Fig. 10, i.e., one is the **equal allocation** and the other is the **application driven allocation**.

For application driven allocation, our intuition is shifting more resource for bandwidth hungry applications so as to improve the overall quality for all users. Ideally, we would like to find out the optimal combination of different users (with different applications) that yields the maximum overall quality $Q_{\mathtt{tot}}$. For different applications, its quality depends on its content

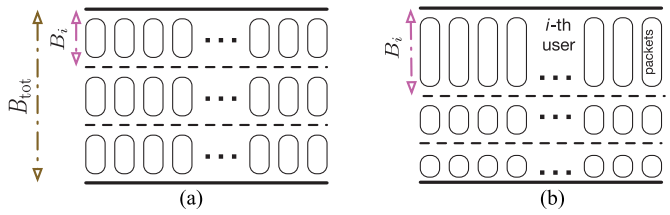


Fig. 10. Multi-user quality-bandwidth optimization: (a) equal allocation; (b) application driven allocation. $B_{\texttt{tot}}$ is the total bandwidth for all users and B_i is the bandwidth allocation for i-th user. Both $B_{\texttt{tot}}$ and B_i are adaptive because of the network fluctuation.

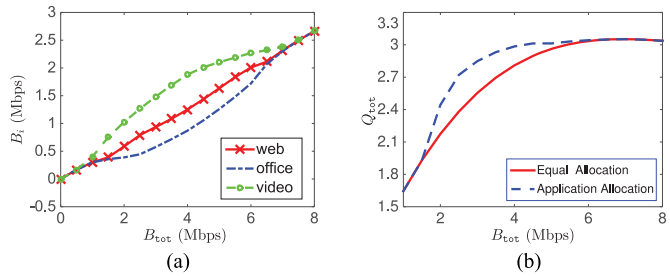


Fig. 11. Application driven bandwidth allocation: (a) individual bandwidth for each user; (b) overall quality vs. total bandwidth.

characteristics and bit rate [43]. Together with (1), we could formulate the optimization problem as

$$\max Q_{\texttt{tot}} = \sum_i Q_i(R_i), \text{subject to} \sum_i R_i \leq \alpha \cdot B_{\texttt{tot}} \quad (2)$$

where $B_{\texttt{tot}}$ is the total bandwidth presumably allocated for all users. However, due to the network dynamics, it is also a variable over the time. $\alpha = 0.9$ is used in this work.

Referring to the typical scenarios aforementioned, i.e., "office", "web" and "video", we have demonstrated the application driven bandwidth allocation as shown in Fig. 11. Here, we assume the maximum network bandwidth at 8 Mbps for these three typical users. Total bandwidth varies due to the network dynamics, such as the wireless channel fading, congestions, etc. Given any instant total bandwidth, we can dynamically allocate optimal bandwidth resources to different users to maximize overall delivered video quality. Specifically, we perform the exhaustive search to solve this problem. For example, we assume the step size of bit rate variation is 100 kbps (resulting in 81 discrete bit rate points from 0 to 8 Mbps). For any given $B_{\texttt{tot}}(t)$, we look up all possibilities that $\sum_i R_i(t) \leq \alpha B_{\texttt{tot}}(t)$ and pick the one yielding the best quality. Even for the worst case, we need to calculate $81 \times 81 \times 81 = 531411$ times. For any mainstream CPU offering Gigahertz frequency, it takes several milliseconds which is negligible.

As we can see, motion intensive content (e.g., video) demands more network bandwidth while stationary content (i.e., office) could be managed with less bandwidth shown in Fig. 11(a) with better overall quality compared with the most straightforward equal allocation strategy depicted in Fig. 11(b). But if the network is too bad or too good, there isn't difference between two allocation mechanisms. This is also consistent with our expectation.

VI. Conclusion

In this paper, we proposed an interactive screen video streaming based system to enable the pervasive mobile workstyle, which is referred to as the PC2PC, where the PC2PC server compresses the desktop screens using HEVC based SCC encoder, and delivers the stream through network to the client for stream decoding, rendering and end-user interaction. Network bandwidth is estimated at client side and reported back to the server side using system message for quality-bandwidth adaptation. Extensive studies have demonstrated the efficiency of the proposed system, by massively reducing the network bandwidth consumption for the same visual quality in comparison to SPICE, by a factor of 2, 7 and 4 respectively for typical video streaming, web browsing and stationary office applications. Gaming experience is close to local console or PC when the network delay is less than 50 ms. Quality-bandwidth tradeoff is studied for both single user and multi-user scenarios, where bandwidth estimation is based on the packet arrival timing measurement at the client. We envision the huge potential of the PC2PC system to be used in our daily life, migrating the computing from the PC era to the true mobile era. We make the PC2PC platform after further refinement available to the public and welcome the opportunities to work with any interested parties.

Acknowledgment

The authors would like to thank all anonymous reviewers for their constructive comments to improve this paper.

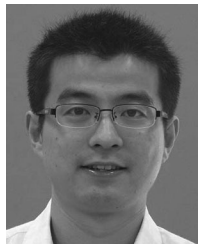
References

[1] Citrix Mobile Workstyles Survey, "Workplace of the future: A global market research report," 2014. [Online]. Available: https://www.citrix.com/content/dam/citrix/en_us/documents/products-solu tions/workplace-of-the-future-a-global-market-research-report.pdf
[2] VMware Brief, "Mobile secure desktop solution: Mobile, secure access to applications and data across devices and locations," 2014. [Online]. Available: https://www.vmware.com/files/pdf/view/Mobile-Secure-Desktop-Solution-Br ief.pdf
[3] "VMware Horizon," 2017. [Online]. Available: http://www.vmware.com/products/horizon-view
[4] "Citrix XenDesktop," 2017. [Online]. Available: https://www.citrix.com/products/xendesktop/overview.html
[5] "Microsoft RDP," 2014. [Online]. Available: https://en.wikipedia.org/wiki/Remote_Desktop_Protocol
[6] "SPICE," 2016. [Online]. Available: http://www.spice-space.org/
[7] "PC2PC," 2017. [Online]. Available: http://vision.nju.edu.cn/index.php/research/item/66-pc2pc
[8] "QEMU," 2017. [Online]. Available: http://wiki.qemu.org/
[9] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," *IEEE Trans. Inf. Theory*, vol. IT-24, no. 5, pp. 530–536, Sep. 1978.
[10] Z. Ma, W. Wang, M. Xu, and H. Yu, "Advanced screen content coding using color table and index map," *IEEE Trans. Image Process.*, vol. 23, no. 10, pp. 4399–4413, Oct. 2014.
[11] Y.-F. Ou, T. Liu, Z. Zhao, Z. Ma, and Y. Wang, "Modeling the impact of frame rate on perceptual quality of video," in *Proc. IEEE Int. Conf. Image Process.*, San Diego, Oct. 2008, pp. 689–692.
[12] K. Chen *et al.*, "On the quality of service of cloud gaming systems," *IEEE Trans. Multimedia*, vol. 16, no. 2, pp. 480–495, Feb. 2014.

[13] Z. Ma, M. Xu, Y.-F. Ou, and Y. Wang, "Modeling rate and perceptual quality of video as functions of quantization and frame rate and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 5, pp. 671–682, May 2012.
[14] L. Fang, O. C. Au, and N. Cheung, "Subpixel rendering: From font rendering to image subsampling," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 177–189, May 2013.
[15] T. Wiegand, G.-J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.
[16] G.-J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.
[17] A. Grange, *Overview of VP-Next*, Internet Engineering Task Force, Fremont, CA, USA, Dec. 2012.
[18] "Audio Video Coding Standards," 2017. [Online]. Available: http://www.avs.org.cn/
[19] T. Lin, P. Zhang, S. Wang, K. Zhou, and X. Chen, "Mixed chroma sampling-rate high efficiency video coding for full-chroma screen content," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 1, pp. 173–185, Jan. 2013.
[20] Y. Lu, S. Li, and H. Shen, "Virtualized screen: A third element for cloud-mobile convergence," *IEEE Multimedia*, vol. 18, no. 2, pp. 4–11, Feb. 2011.
[21] Z. Pan, H. Shen, Y. Lu, and S. Li, "A low-complexity screen compression scheme for interactive screen sharing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 6, pp. 949–960, Jun. 2013.
[22] S. Wang *et al.*, "Utility-driven adaptive preprocessing for screen content video compression," *IEEE Trans. Multimedia*, vol. 19, no. 3, pp. 660–667, May 2017.
[23] ISO/IEC JTC1/SC29/WG11 MPEG, *Joint Call for Proposals for Coding of Screen Content*, MPEG N14715, Jan. 2014.
[24] D.-K. Kwon and M. Budagavi, "RCE3: Results of Test 3.3 on intra motion compensation," Joint Collaborative Team on Video Coding, Doc. JCTVC-N0205, Jul. 2013.
[25] C. Pang, J. Sole, L. Guo, M. Karczewicz, and R. Joshi, "Non-RCE3: Intra motion compensation with 2-D MVs," Joint Collaborative Team on Video Coding, Doc. JCTVC-N0256, Jul. 2013.
[26] J. Chen *et al.*, "Description of screen content coding technology proposal by Qualcomm," Joint Collaborative Team on Video Coding, Doc. JCTVC-Q0031, Apr. 2014.
[27] L. Guo, M. Karczewicz, and J. Sole, "RCE3: Results of Test 3.1 on palette mode for screen content coding," Joint Collaborative Team on Video Coding, Doc. JCTVC-N0247, Jul. 2013.
[28] W. Zhu, J. Xu, and W. Ding, "RCE3 Test 2: Multi-stage base color and index map," Joint Collaborative Team on Video Coding, Doc. JCTVC-N0287, Jul. 2013.
[29] Z. Ma, W. Wang, M. Xu, X. Wang, and H. Yu, "Description of screen content coding technology proposal by Huawei Technologies (USA)," Joint Collaborative Team on Video Coding, Doc. JCTVC-Q0034, Apr. 2014.
[30] W. Pu, X. Guo, P. Onno, P. Lai, and J. Xu, "AHG10: Suggested software for palette coding based on RExt6.0," Joint Collaborative Team on Video Coding, Doc. JCTVC-Q0094, Apr. 2014.
[31] B. Li, J. Xu, G. Sullivan, Y. Zhou, and B. Lin, "Adaptive motion vector resolution for screen content," Joint Collaborative Team on Video Coding, Doc. JCTVC-S0085, Oct. 2014.
[32] J. Boyce, "BoG report on Adaptive Color Transform (ACT)," Joint Collaborative Team on Video Coding, Doc. JCTVC-S0304, Oct. 2014.
[33] "libyuv," 2017. [Online]. Available: https://chromium.googlesource.com/libyuv/libyuv/
[34] X. Feng, F Yang, H. Zhang, N. Shi, and Z. Ma, "On video source format of screen content compression," in *Proc. IEEE Visual Commun. Image Process.*, Dec. 2015, pp. 1–4.
[35] "FFmpeg," 2017. [Online]. Available: http://www.ffmpeg.org/
[36] J.-R. Ohm, G.-J. Sullivan, H. Schwarz, T.-K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—Including high efficiency video coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec. 2012.
[37] I.-K. Kim, J. Min, T. Lee, W.-J. Han, and J. Park, "Block partitioning structure in the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1697–1706, Dec. 2012.
[38] H. Yaïche, R. R Mazumdar, and C. Rosenberg, "A game theoretic framework for bandwidth allocation and pricing in broadband networks," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 667–678, Oct. 2000.
[39] C. M Assi, Y. Ye, S. Dixit, and M. A. Ali, "Dynamic bandwidth allocation for quality-of-service over Ethernet PONs," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 9, pp. 1467–1477, Nov. 2003.
[40] "Firefly RK3288," 2016. [Online]. Available: http://en.t-firefly.com/en/
[41] Y. Hu, D. Niu, and Z. Li, "A geometric approach to server selection for interactive video streaming," *IEEE Trans. Multimedia*, vol. 18, no. 5, pp. 840–851, May 2016.
[42] "Clumsy," 2016. [Online]. Available: http://jagt.github.io/clumsy/index.html
[43] H. Hu, Z. Ma, and Y. Wang, "Optimization of spatial, temporal and amplitude resolution for rate-constrained video coding and scalable video adaptation," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2012, pp. 717–720.

Zhan Ma (S'06–M'11) received the B.S. and M.S. degrees from Huazhong University of Science and Technology, Wuhan, China, in 2004 and 2006, respectively, and the Ph.D. degree from the Tandon School of Engineering of New York University (formerly Polytechnic University, Brooklyn, NY, USA), New York, NY, USA, in 2011. He is currently on the faculty of Electronic Science and Engineering School, Nanjing University, Nanjing, China. From 2011 to 2014, he was with Samsung Research America, Dallas, TX, USA, and Futurewei Technologies, Inc., Santa Clara, CA, USA, respectively. His current research interests include the video compression, gigapixel streaming, and multispectral signal processing.

Tao Yue received the B.S. degree in automation from Northwestern Polytechnical University, Xi'an, China, in 2009, and the Ph.D. degree from Tsinghua University, Beijing, China, in 2015. He is currently an Associate Professor with the School of Electronic Science and Engineering, Nanjing University, Nanjing, China. His research interests mainly include computer vision, image processing, and computational photography.



Xun Cao (S'10–M'12) received the B.S. degree from Nanjing University, Nanjing, China, in 2006, and the Ph.D. degree from the Department of Automation, Tsinghua University, Beijing, China, in 2012. He held visiting positions with Philips Research, Aachen, Germany, in 2008, and Microsoft Research Asia, Beijing, from 2009 to 2010. He was a Visiting Scholar with the University of Texas at Austin, Austin, TX, USA, from 2010 to 2011. He is currently a Professor with the School of Electronic Science and Engineering, Nanjing University. His research interests include computational photography, image-based modeling and rendering, and VR/AR systems.

Yiling Xu received the B.S., M.S., and Ph.D. degrees from the University of Electronic Science and Technology of China, Chengdu, China, in 1999, 2001, and 2004, respectively. She is currently a Full Researcher with the School of Electronic Information and Electronic Engineering, Shanghai Jiaotong University, Shanghai, China. From 2004 to 2013, she was with Multimedia Communication Research Institute of Samsung Electronics, Inc., Korea. Her research interests mainly include architecture design for next generation multimedia systems, dynamic data encapsulation, adaptive cross layer design, dynamic adaption for heterogenous networks, and N-screen content presentation.

Xin Li received the B.S. and M.S. degrees from Huazhong University of Science and Technology, Wuhan, China, in 2003 and 2005, respectively, and the M.S. degree from the Volgenau School of Engineering of George Mason University, Fairfax, VA, USA, in 2013. He is currently the CTO of a startup company with focuses on cloud gaming and gigapixel streaming. From 2012 to 2015, he was with Samsung Research America working on Green MPEG, MicroServer, and Samsung Virtual Reality platform. From 2005 to 2008, he worked at Intel China Software Development Center at Shanghai on UEFI—The next generation BIOS.

Yongjin Wang received the Ph.D. degree from Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai, China, in 2005. He held various research positions in Germany, Japan, and the U.K. with the support of an Alexander von Humboldt research fellowship, a Japan Society for the Promotion of Science research fellowship, and the Royal Academy of Engineering research fellowship, respectively. Since 2011, he has been a Professor with Nanjing University of Posts and Telecommunications, Nanjing, China.