JAVRE: A Joint Asymmetric Video Rendering and Encoding Approach to Enable Optimized Cloud Mobile 3D Virtual Immersive User Experience

Yao Lu, *Student Member, IEEE*, and Sujit Dey, *Fellow, IEEE*

***Abstract*— With the growing adoption of mobile devices as the primary device for information and media consumption, and recent growth in availability of mobile devices with autostereoscopic 3D displays, the time has come to enable rich 3D virtual immersive experiences on mobile devices, like 3D display gaming, 3D virtual classroom, 3D virtual art gallery, etc., which are currently enabled on PC platforms. Instead of developing such compute-intensive applications on mobile devices, a scalable and portable approach is to take advantage of the elastic cloud resources. In this paper, we explore the possibility of developing a cloud based mobile 3D virtual immersive application architecture, where the 3D video rendering and encoding are performed on cloud servers, with the resulting 3D video streamed to mobile 3D devices through wireless networks. However, with the significantly higher bit rate requirement for 3D videos, ensuring user experience is a challenge considering the inherent dynamic variability of mobile networks. In this paper, in order to address this challenge, we propose a joint asymmetric graphics rendering and video encoding approach called JAVRE, where both the encoding quality and rendering richness of the left view and the right view can be asymmetric. Specifically, we first develop two mathematical models (user experience model and video bit rate model) through subjective and objective tests. Then we propose an optimization algorithm that can automatically choose the video encoding settings and graphics rendering settings for the left view and the right view to ensure the best user experience given the network conditions. Experiments conducted using real 4G-LTE network profiles and a commercial cloud service demonstrate significant improvement in user experience when JAVRE is used.**

***Index Terms*— 3D, Cloud mobile virtual immersive application, user experience.**

I. Introduction

CLOUD mobile 3D display gaming [CMG(3D)] architecture [1], [2] was proposed in recent years as an extension of cloud mobile gaming (CMG) architecture [3]–[6] to enable true 3D immersive gaming experience for 3D devices. Different from the traditional standalone game architecture in which the games are downloaded and rendered on user's local mobile devices, the CMG(3D) architecture allows the game to be rendered on a remote server in order to enable rich 3D gaming experience on thin mobile devices with very limited battery consumption. Cloud mobile 3D virtual immersive application [CMVIA(3D)] architecture, on the other hand, is very similar to CMG(3D) architecture. The difference is that it not only supports games, but it also supports much more 3D virtual immersive applications, such as virtual classroom [7], virtual art gallery [8], virtual tourism [9], etc. The system architecture diagram is shown in Fig. 1. In detail, we place two virtual cameras in the virtual world to generate a left view and a right view of the virtual scene. After the two views are generated, they will be captured from the screen and encoded as 3D videos and transmitted through wireless network to the mobile device, and displayed on the device 3D screen. On the reverse side, control commands are captured and transmitted from mobile device to the server to execute. The CMVIA(3D) architecture has the advantage of low power and low storage consumption on the mobile devices as well as cross-platform ability which saves tremendous effort for 3D application developers while it shifts the challenge from the client to the server on how to stream high quality 3D video through fluctuating network conditions with low latency.

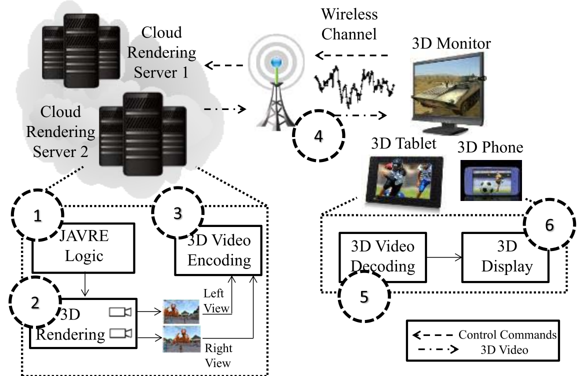


Fig. 1. System architecture diagram of CMVIA(3D).

To address the above problem, asymmetric video encoding techniques, which dynamically set quantization parameter/resolution differently for two views, can be potentially used to lower bit rate while minimizing impact on user experience [10], [11]. More recently, asymmetric graphics rendering, which dynamically sets rendering parameters like

Manuscript received February 20, 2016; revised May 24, 2016 and July 16, 2016; accepted August 18, 2016. Date of publication September 8, 2016; date of current version December 9, 2016. This material is based upon work supported partly by the National Science Foundation under Grant CCF-1160832, and partly by the Center for Wireless Communications. This paper was recommended by Guest Editor J. Xu.

The authors are with the Mobile System Design Lab, Department of Electrical and Computer Engineering, University of California-San Diego, La Jolla, CA 92093 USA (e-mail: luyao@ucsd.edu; dey@ece.ucsd.edu).

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

Digital Object Identifier 10.1109/JETCAS.2016.2602246

texture detail and/or view distance differently for two views, has been proposed to lower bit rate while attempting to preserve user experience [1], [2]. However, previously the above techniques are studied and applied separately. In this paper, we explore the possibility of combining asymmetric video encoding and asymmetric graphics rendering techniques together to jointly optimize the whole system and therefore to provide the best user experience. To quantitatively measure the user experience by different video encoding settings and graphics rendering settings, we first perform subjective tests and develop a user experience model. Second, we also develop a bit rate model to estimate the video bit rate value as a function of video encoding settings and graphics rendering settings. By making use of the above two models, we further propose an optimization algorithm called *JAVRE* to automatically choose the video encoding settings and the graphics rendering settings for the left and right views to ensure the best overall user experience given the network conditions. Experiments performed on Amazon Cloud Services demonstrate up to 64.5% user experience increase over the existing methods.

The rest of the paper is organized as following. Section II reviews related work. Section III briefly introduces our over system diagram. In Section IV, we propose a user experience model to quantitatively measure the user experience according to different video encoding settings and graphics rendering settings. In Section V, we propose another model to estimate the video bit rate as a function of video encoding settings and graphics rendering settings. Section VI proposes *JAVRE* to ensure the best overall user experience. Section VII shows the experimental results using our CMVIA(3D) prototype hosted on Amazon Web Service. Section VIII proposes future work and concludes the paper.

II. Related Work

This work overlaps research from the following four main areas: video coding technology (Section II-A), user experience modeling based on subjective tests (Section II-B), video bit rate modeling (Section II-C) and adaptive encoding and rendering techniques (Section II-D).

A. Video Coding Technology

For 3D virtual immersive applications, the main difference between the traditional stand alone rendering architecture versus the cloud based rendering architecture is that the latter uses video encoding technology to compress the screen content as a video and then transmit. Upgrading video codec to the latest standard is the easiest way to reduce bit rate consumption and therefore reduce delay and improve user experience.

Video coding standards have evolved primarily through the development by ITU-T and ISO/IEC standardization groups. H.261 [12] and H.263 [13] are produced by ITU-Twhile MPEG-1 [14] and MPEG-4 Visual [15] are produced by ISO/IEC. The two organizations jointly produced the H.262/MPEG-2 Video [16], H.264/MPEG-4 Advanced Video Coding (AVC) [17] and H.265/MEPG-H High Efficiency Video Coding (HEVC) [18] standards. The latest released standard HEVC is claimed to be able to encode a video using half of the bit rate needed by AVC while producing the same quality. There are various extensions of HEVC that are being developed and released afterwards among which there is an extension called Screen Content Coding (SCC) [19]. SCC is designed to improve compression capability for video containing a significant portion of rendered (moving or static) graphics, text, or animation rather than camera-captured video scenes.

However, even if the bit rate can be reduced by using new standards, it may still not meet bandwidth constraint and therefore we develop techniques that can be used combined with the advantage of upgrading codec. In this paper we use an open source implementation of HEVC standard without SCC extension (since it is not finalized and not implemented) called x265 [20] version 1.8 as the codec.

B. User Experience Model Based on Subjective Tests

Vankeirsbilck et al. [21] propose a user experience model for cloud gaming but it only considers video encoding impairment and it is based on a single view 2D display CMG system. Liu et al. [22] have done user experience study considering both video encoding impairment and graphics rendering impairment. However, it is also applicable only to 2D display CMG system. In [2], we considered modeling user experience for CMG(3D). However, in that work, only graphics rendering impairment is considered. In this work, we aim to develop a user experience model not only for gaming, but also for a variety of 3D virtual immersive applications which considers the joint impairment caused by video encoder and graphics rendering engine.

C. Video Bit Rate Model

Several models have been proposed to model the relationship between video bit rate and encoding settings/video content features [23], [24]. However, there are several shortcomings. 1) These models are not specifically designed for screen content videos. 2) These models do not take into account graphics rendering settings. 3) These models are developed several years ago with videos of low resolutions. 4) Some models are developed based on old standards. In our previous work [2], we have also tried to model the video bit rate as a function of graphics rendering settings. However, the model is based on AVC standard and also because it does not include video encoding settings, it has to work only under high motion assumption. In this work, we propose a new bit rate model based on HEVC standard which takes into account both graphics rendering settings and video content features so that it is more general and accurate.

D. Adaptive Encoding and Rendering Techniques

Adaptive video encoding techniques, specifically for 3D videos, have been proposed previously. Asymmetric 3D video encoding adaptation [10], [11], where the videos of the left and right views are encoded with different quality and can adapt in response to network bandwidth constraints, has been proved to have 20%–40% bit rate savings without additional impairment. Adaptive graphics rendering techniques for CMG(3D) are also proposed in our previous work [2] in which the rendering engine chooses different texture detail or view distance settings for the left view and the right view in a way that reduces the bit rate of the resulting encoded video in order to meet the network constraint. However, none of the above work has considered the possibility of optimizing the whole system by applying both asymmetric video encoding adaptation and asymmetric graphics rendering adaption at the same time. In this paper, we research this combined way of adaptation to further save 3D video bit rate while preserving the quality.

III. System Overview

Fig. 1 shows our overall system diagram. In step 1, our algorithm will decide the best texture detail (TD) and quantization parameter (QP) for both left and right views. Then, in step 2, the graphics engine will generate (render) two original views according to the TD decided in step 1. In step 3, the original views will be encoded as a bit-stream using the QP decided in step 1. In step 4, the encoded bit-stream will be transmitted through the wireless network to the user's client device. In step 5, the device will decode the bit-stream to recover the video frames. In step 6, the views for left eye and right eye will be displayed on the 3Ddevice. Note that in the proposed system, the video frames encoded, transmitted, decoded and displayed are all of the same resolution that is 960×720. Thus, compared to the traditional video encoding problem in which the original view is given and only encoding parameters (resolution, QP, framer ate, etc.) can be changed to achieve the best rate-distortion performance, our system offers one more degree of freedom: to generate (render) the original view itself. By changing rendering parameters (TD, view distance, etc.) it influences how the original views are generated. Note that in this way, the quality of the final views that the user sees depend on both how the original views are generated (step 2) and how much distortion is caused by encoding (step 3). We therefore conduct user experience experiments to propose a model to derive a final user experience score according to both rendering parameters (TD) and encoding parameters (QP).

IV. User Experience Model

In this section, we study and model user experience for CMVIA(3D) considering the joint effect of video encoding impairment and graphics rendering impairment, as opposed to our previous work [2] which only takes into account the effects of graphics rendering impairment for CMG(3D). Because both the system architecture and the factors that influence user experience of CMVIA(3D) and CMG(3D) are very similar, we use the same Mean Opinion Score (*MOS*) criterion introduced in [2] as a measurement metric for modeling CMVIA(3D) user experience, as shown in

$$\mathrm{MOS} = 1 + 0.035R + 7 \times 10^{-6} R\,(R - 60)\,(100 - R) \quad (1)$$

$$R = 100 - I\,(VE, GR) \quad (2)$$

In (1), the *MOS* metric is formulated by a transmission rating factor R, which represents the overall user experience. R factor takes value between 0 and 100; higher R value means higher *MOS* and better user experience. In (2), the equation I (*VE, GR*) stands for the combined impairment caused by video encoding (*VE*) and graphics rendering (*GR*).

The proposed *MOS* metric (1) is actually the same as is proposed by ITU-T E-Model [38]. However, we redefine the formulation of R (2) to best describe the factors that need to be considered for CMVIA(3D).

In the following, we describe how we conducted the subjective tests to derive the impairment functions and how we validated them.

TABLE I
EXPERIMENT SETTING

Settings	*Experiment Values*
Texture Detail(Down Sample)	High(0) Medium(1) Low(2)
Quantization Parameter	25 27 29 31 33 35 37 39

TABLE II
x265 ENCODING SETTINGS

Encoding Parameters	*Value*	*Encoding Parameters*	*Value*
Rate Control	CQP	Number of Reference frames	3
Profile	Main	Number of B frames	0
Preset	Medium	Scenecut	40
Rc-lookahead	0	Period Intra Refresh	ON
Wavefront Parallel Processing	ON	Weighted P Frame	ON
AMP Partition	ON	Weighted B Frame	OFF
Motion Estimation Range	57	Enable PSNR	ON

A. Subjective Test settings

Table I shows the specific video encoding settings and graphics rendering settings we want to include in our experiment. In detail, for video encoding settings, we fix the resolution to be 960×720 for each view and frame rate to be 25 fps but change the quantization parameter (*QP*). *QP*, which ranges from 0 to 51, decides the quantization level used. Higher *QP* means lower quality. The other settings of x265 encoder are listed in Table II. For graphics rendering settings, we study the effect of asymmetric texture detail. Texture detail defines the quality of the images on the surface of the objects. As is defined in [2], we define texture detail to be high when the game is using the original texture images, to be medium when the texture images are downsampled once and low when the texture images are downsampled twice. Fig. 3(c) shows an example where the left view is rendered with high quality and the right view is rendered with medium quality. The viewer can observe some blurry effect from the right view.

Fig. 2 shows the testbed used for the subjective tests. We use a 3D monitor with a laptop to substitute for 3D display of mobile devices because current available mobile 3D displays do not have as good quality as 3D monitors that may cause

Fig. 2. Testbed for subjective experiments.

TABLE III
IMPAIRMENT CRITERION

I	*0*	*0-20*	*20-40*	*40-60*	*60-100*
Description (Impairment)	No	Minor	Noticeable	Clear	Unacceptable

additional impairment that we want to avoid. The laptop is connected to an access point and the access point is connected to the game server. The selected applications which run upon the above framework are 1) an online open-source MMORPG game PlaneShift [25], 2) a virtual classroom application we developed based on SecondLife [26], and 3) a virtual art gallery application we also developed based on SecondLife. We then invited 25 students (17 male, 8 female; aged 18–26) to participate in our subjective experiments. Firstly, we asked each tester to sit before a 23-in LG D2342 3D Monitor, and view a 3D video as a training sequence before the real test starts to let each tester adjust their viewing angle optimally. We also teach the testers how to use keyboard to control the avatar, who is NPC and what to look for in the virtual world. After that, we start these 3D applications and manually set the video encoding settings and graphics rendering settings according to Table I independently for each view. For each application, we start by setting *TD* to be High and *QP* to be 25 for both views. We keep *QP* for the right view to be 25 but increase *QP* for the left view one at a time till it reaches 39. The above is done for each *TD* combination. After that, we keep *QP* for the left view to be 25 but change *QP* of the right view and repeat the experiments for each *TD* combination. Once a combination of rendering and encoding factors is set, we ask the testers to play the game for 1 min. The testers are given a pencil and a table to fill in. At the end of each condition, we will stop the application. The testers will remove the glasses, evaluate the impairment according to the criterion listed in Table III, fill in the scores using the pencil and wear the glasses again to do the experiment for the next round. During the whole experiment, the testers were asked to control the avatar to perform multiple tasks (including attacking an enemy in the gaming application, discussing with another student in the virtual classroom application, watch the paintings in the virtual art gallery application, etc.). Example snapshots of the three applications with specific graphics rendering and video encoding settings are shown in Fig. 3.

B. Impairment Function Derivation

Considering the graphics rendering and the video encoding settings used in our CMVIA(3D) platform and the two views,

TABLE IV
T(TD) FOR DIFFERENT TEXTURE DETAIL SETTINGS

Application	H	M	L
Gaming	27	31	35
Virtual Classroom	27	31	35
Virtual Art Gallery	27	29	31

TABLE V
AVERAGE I_{TD} FOR DIFFERENT TEXTURE DETAIL COMBINATIONS

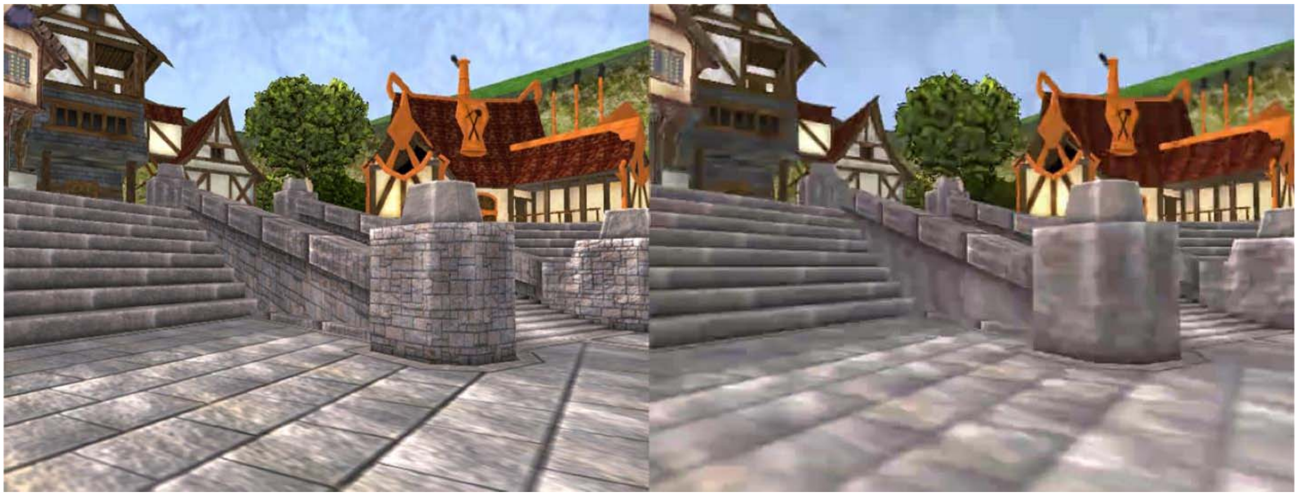
Application	H-H	H-M	M-M	M-L	L-L
Gaming	0.2	8.35	12.3	19.14	24.5
Virtual Classroom	0	7.4	11.5	17.2	22.6
Virtual Art Gallery	0	11.2	15.7	24.3	32.3

we formulate impairment function as

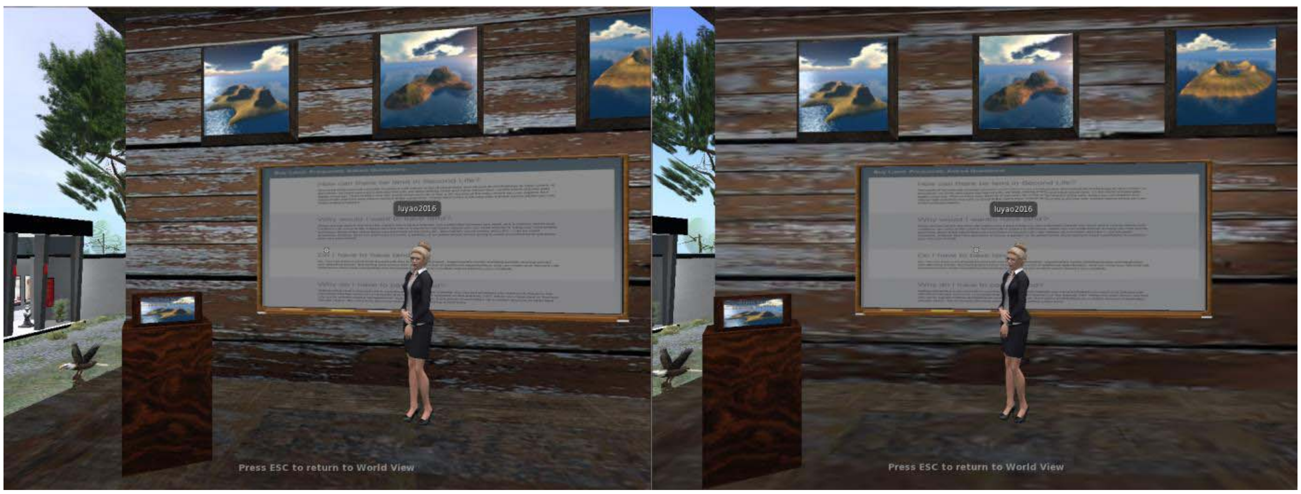
$$I\,(VE, GR) = I\,(TD_L, TD_R, QP_L, QP_R) \tag{3}$$

where *TD* means texture detail and *QP* indicates quantization parameter. The subscripts *L* and *R* represent left view and right view respectively. In order to study $I(VE, GR)$, we first keep one of the four parameters fixed to its best quality value during the test and see how the impairment *I* changes according to the other three parameters.

Fig. 4(a)–(f) shows the average impairment values when we keep *QP* of one of the views to be 25 (almost no video encoding impairment on that view) but change *QP* of the other view and at the same time change the texture detail settings for both views. From these six figures we can clearly observe that for each texture detail combination, the impairment values remain similar till the QP value exceeds some thresholds (showing as a red circle on the figures). For example, for curve *L-L* (both left and right views use low texture detail) in Fig. 4(a), the threshold is at 35 since the impairment *I* does not change until QP exceeds 35. Clearly the value of this QP threshold is related to the texture detail setting. Further, we found that this threshold was only related to the texture detail setting of the view whose *QP* is changing. For example, in Fig. 4(a), the threshold of M-M is the same as the threshold of M-L and that value is also the same as the threshold of H-M and M-M in Fig. 4(b). Thus, it means for a specific texture detail setting of one view, there is a corresponding *QP* threshold so that when *QP* is less than or equal to that threshold, the total impairment is not related to the *QP* value. In other words, the video encoding will not cause additional impairment besides the impairment caused by texture detail when the QP is below the threshold. According to the results from Fig. 4(a)–(f), we list the threshold and the corresponding impairment I_{TD} in Tables IV and V. Note that although the above relationship as well as all the models derived subsequently are general, the values of the parameters need to be derived for each specific application. From Tables IV and V we can also identify that the virtual art gallery application has much higher impairment due to texture detail and lower *QP* threshold than the other two applications. The reason is that



(a)



(b)

(c)

Fig. 3. Example snapshots for three applications (a) top, cloud mobile 3D display gaming left view: High texture detail and $QP = 25$, right view: Medium texture detail and $QP = 35$, (b) middle, cloud mobile 3D virtual classroom left view: High texture detail and $QP = 25$, right view: Medium texture detail and $QP = 27$, (c) bottom, cloud mobile 3D virtual art gallery left view: High texture detail and $QP = 25$, right view: Medium texture detail and $QP = 25$.

for this application, the user will pay continuous attention to the content of the virtual paintings so that they tend to be very strict about the details of the views. Thus, when there is some blurry effect of the views, it hurts user experience severely.

Considering the QP threshold, we propose to model I by two parts. The first one is the I_{TD} that is the impairment caused by texture detail only and the second part is I_A that is the additional impairment when QP is bigger than the

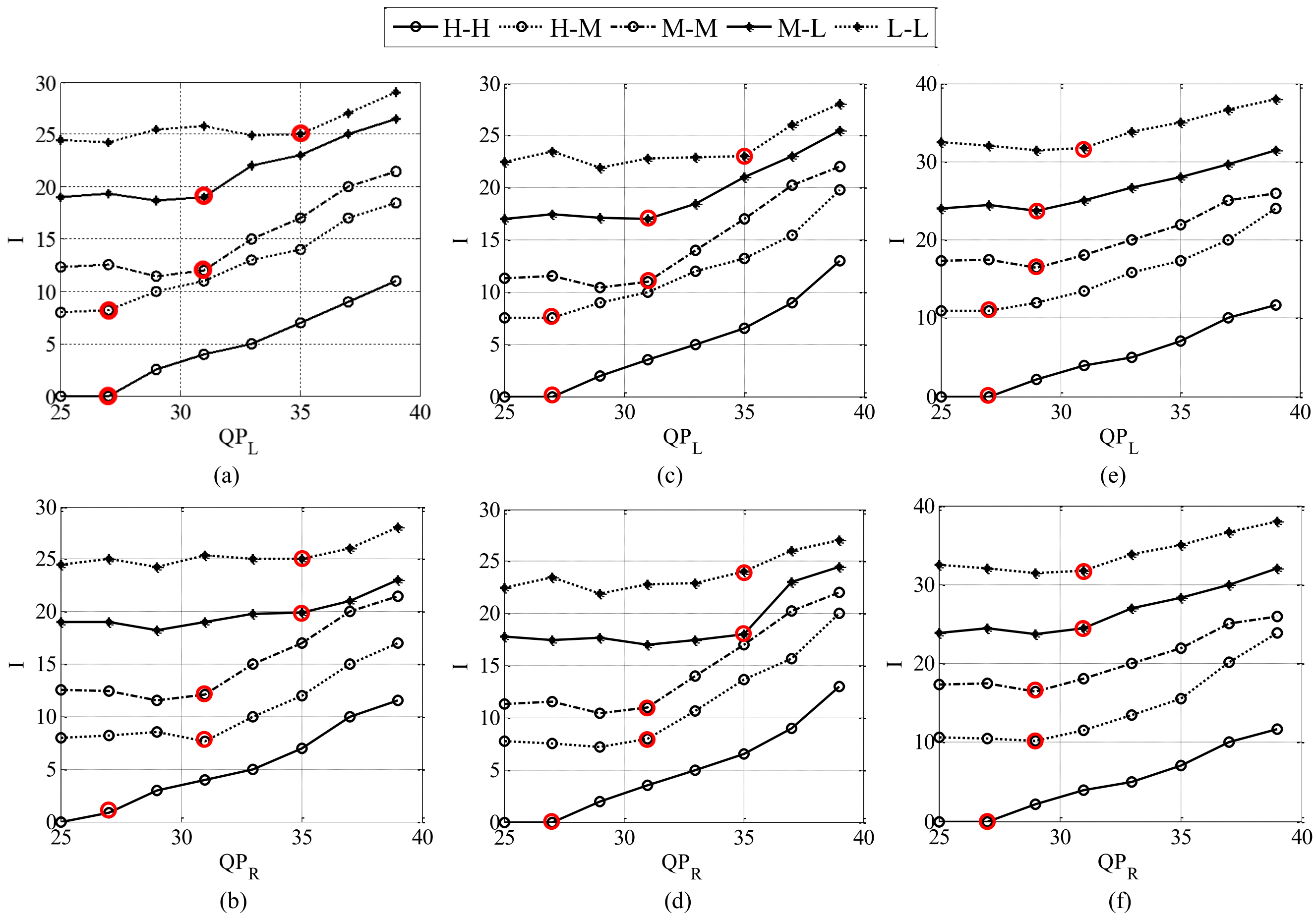


Fig. 4. (a) top left, Relationship between I and QP_L under different texture detail combinations for gaming (b) bottom left, Relationship between I and QP_R under different texture detail combinations for gaming (c) top middle, Relationship between I and QP_L under different texture detail combinations for virtual classroom (d) bottom middle, Relationship between I and QP_R under different texture detail combinations for virtual classroom (e) top right, Relationship between I and QP_L under different texture detail combinations for virtual art gallery (f) bottom right, Relationship between I and QP_R under different texture detail combinations for virtual art gallery.

threshold. Equation (4) shows the relationship

$$I\left(TD_L, TD_R, QP_L, QP_R\right) = I_{TD}\left(TD_L, TD_R\right) + I_A$$

$$I_A = \begin{cases} 0 & QP_L < T(TD_L) \\ & QP_R < T(TD_R) \\ f\left(QP_L - T(TD_L)\right) & QP_L \geq T(TD_L) \\ & QP_R < T(TD_R) \\ f\left(QP_R - T(TD_R)\right) & QP_L < T(TD_L) \\ & QP_R \geq T(TD_R) \\ g\left(QP_L - T(TD_L),\right. & \\ \quad \left. QP_R - T(TD_R)\right) & QP_L \geq T(TD_L) \\ & QP_R \geq T(TD_R) \end{cases} \tag{4}$$

in which I_{TD} is the value form Table V. The QP threshold *T(TD)* is the value from Table IV.

From Fig. 4(a)–(f) we observe that the impairment I would increase almost linearly with QP when QP exceeds the threshold *T(TD)*. Hence we can conclude that $f()$ can be modeled as a linear function showing in

$$f\left(QP_L - T(TD_L)\right) = a\left(QP_L - T(TD_L)\right) + b \tag{5}$$

TABLE VI
REGRESSION PARAMETERS OF a AND b FOR f FUNCTION

Application	a			b		
	H	M	L	H	M	L
Gaming	0.9	0.8	1.1	0.1	0.3	0.2
Virtual Classroom	0.95	0.88	1.03	0.4	0.2	0.25
Virtual Art Gallery	0.88	0.75	1.21	0.3	0.23	0.14

The parameters a and b for three applications are derived using linear regression technique and are listed in Table VI.

In order to derive the $g()$ function, we change both *QP* s to be values that exceeds *T(TD)*, plot the data points in a 3D figure and derive the relationship. Fig. 5 shows an example for gaming application with texture detail combination set to be High-High. We tried different two dimensional equations to fit the data and we found that the bilinear equation can have the best regression results in terms mean square error (MSE). Thus we model the $g()$ function as is shown in (6). The parameters a_1, a_2, and b_1 for three applications are

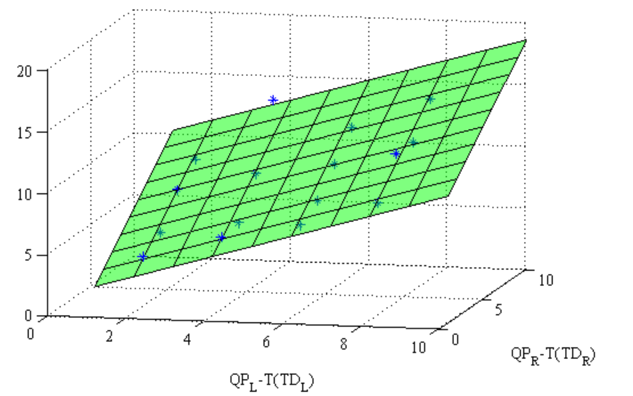


Fig. 5. Regression result for $g()$ function for gaming application when texture detail combination is High-High.

TABLE VII
PARAMETERS OF a_1, a_2 AND b_1 FOR g FUNCTION

Application	H-H	H-M	M-M	M-L	L-L
			a_1		
Gaming	0.92	0.92	0.85	0.82	1.04
Virtual Classroom	0.98	0.96	0.91	0.90	1.13
Virtual Art Gallery	0.91	0.88	0.79	0.76	1.24
			a_2		
Gaming	0.92	0.83	0.85	1.01	1.04
Virtual Classroom	0.98	0.89	0.91	1.05	1.13
Virtual Art Gallery	0.91	0.72	0.79	1.23	1.24
			b_1		
Gaming	0.13	0.12	0.27	0.3	0.22
Virtual Classroom	0.41	0.43	0.22	0.23	0.25
Virtual Art Gallery	0.32	0.34	0.24	0.23	0.15

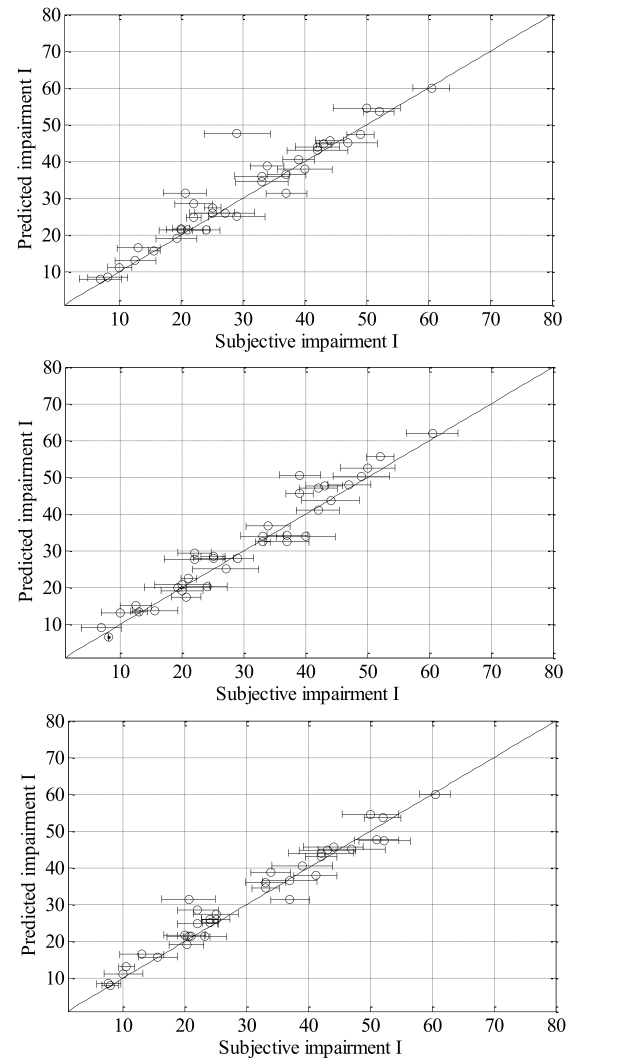


Fig. 6. Validation of $I(VE, GR)$ (a) top, results for gaming (b), middle results for virtual classroom (c) bottom, results for virtual art gallery.

shown in Table VII

$$g\left(QP_L - T\left(TD_L\right), QP_R - T\left(TD_R\right)\right) = a_1\left(QP_L - T\left(TD_L\right)\right) + a_2\left(QP_R - T\left(TD_R\right)\right) + b_1 \quad (6)$$

Equations (3)–(6) complete our user experience model. We will validate it in the next subsection.

C. Model Validation

In order to validate the impairment model [(3) –(6)] derived in the previous subsections, we conducted another set of experiments with a new group of 16 participants (10 male, 6 female; aged 18–25), exploring the same applications. In this set of experiments, the texture detail and quantizationparameters for both views are changed at the same time. Fig. 6 shows the relationship between predicted impairment I computed by the derived impairment function (y-axis) and subjective impairment I given by human subjects (x-axis) for the three applications. In the figures, each data point represents one combination of graphics rendering settings and video encoding settings. We also plotted 95% confidence interval for each measurement as black lines in the figures to show the variety among different subjects. The correlation for the game application is 0.96 while it is 0.97 for virtual classroom application and 0.97 for virtual art gallery application. The above results show the accuracy of the derived user experience model, and its applicability to different applications.

V. BIT RATE MODEL

In TCP based cloud streaming applications, when video bit rate exceeds the network bandwidth, it will cause accumulated delay [31]. Therefore a bit rate model needs to be developed so that given a combination of parameters (TD_L, TD_R, QP_L, QP_R), the video bit rate can be accurately estimated. In this way, by controlling the parameter settings, the resulting video bit rate can be controlled to be below the network bandwidth and therefore avoid congestion and hence delay.

Several techniques have been proposed to model the bit rate of the encoded video as a function of the video encoding parameters. Ma et al. [23] proposed (7) to model bit rate R

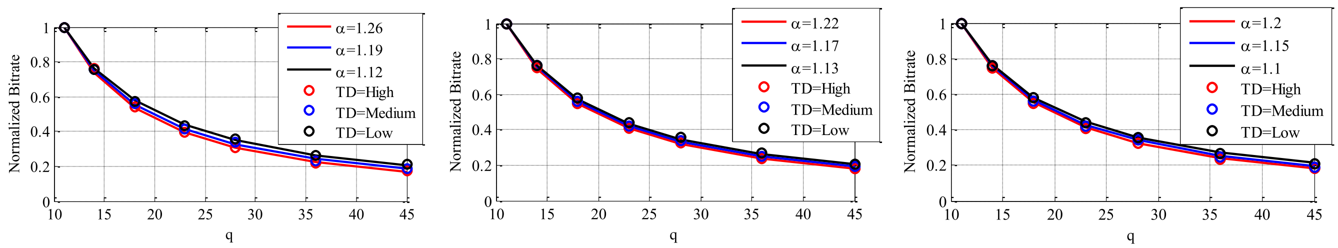


Fig. 7. Validation of model equation (a) left, for gaming; (b) middle, for virtual classroom; (c) right, for virtual art gallery.

using quantization step q and video frame rate t

$$R(q,t) = R_{\max}\left(\frac{q}{q_{\min}}\right)^{-\alpha}\left(\frac{t}{t_{\max}}\right)^{\beta} \tag{7}$$

In (7), coefficients q_{min} and t_{max} represent the minimum quantization step and the maximum frame rate, respectively, and are chosen based on the applicationR_{max} indicates the maximum bit rate when encoding a video at q_{min} and t_{max}; coefficients α and β are model parameters that depend on the content of the video. The authors in [23] further proposed a method to estimate α and β based on content features shown in

$$[\alpha \quad \beta]^T = B\left[1 \quad \mu_{FD} \quad \mu_{MVM} \quad \frac{\mu_{MVM}}{\sigma_{MDA}}\right]^T \tag{8}$$

$$B = \begin{bmatrix} 1.1406 & -0.0330 & -0.0611 & 0.1408 \\ 0.4462 & 0.0112 & 0.0680 & -0.0667 \end{bmatrix} \tag{9}$$

in which μ_{FD} represents mean of frame difference, μ_{MVM} stands for mean of motion vector magnitude and σ_{MDA} means the average of the standard deviation of motion vector directions in each frame.

Though reported in [23] that this model is of high accuracy, it is based on a data set containing videos that are natural scene videos and the resolutionsare CIF (352 × 288) rather than the situation in our CMVIA(3D) application where the view is generated by computer instead of camera in the real world and the video resolution for each view is 960 × 720. Moreover, the video encoding standard we use is H.265/ HEVC instead of H.264/AVC in [23]. Also note that we need to consider both video encoding settings and graphics rendering settings in the bit rate model. Thus, the model (7)–(9), especially the model parameter may not be accurate enough in our case. Therefore, in this paper we extend their work by:

1) Performing experiments using CMIVA(3D) videos with 720 p resolution to validate the model equations.

2) Performing additional experiments to adjust the model of parameter α [(8), (9)] by incorporating graphics rendering setting with H.265/ HEVC video coding standard.

A. Model Equation Validation

In this subsection, we introduce how we perform experiments to validate the model equations. Firstly, because in this paper, we do not evaluate the influence of framerate to user experience or bit rate, we will fix the framerate and set t to be t_{max} and thus we can simplify (7) to be

$$R(q) = R_{\max}\left(\frac{q}{q_{\min}}\right)^{-\alpha} \tag{10}$$

In order to derive and validate this bit rate model, we captured three videos for each application using our CMVIA(3D) system with different texture detail settings and different QP settings from Table I. Using the H.265/HEVC standard definition that $q = 2$ˆ((QP -4)/6) [23], the corresponding q values are 11, 14, 18, 23, 28, 36, and 45. For each video, we encode it by using ×265 encoding library and record the bit rate under each q value. We set R_{max} to be the bit rate when encoding with q_{min} and calculate normalized bit rate $R(q)/R_{max}$. Fig. 7(a)–(c) shows the results. X-axis of the figures is $q$which ranges from 11 to 45 and y-axis is the normalized bit rate. The results of each video in each figure are represented by a specific color. Besides the bit rates shown as circles for the three videos with different texture details, we also plot a line for each video to represent the model equation. The parameter α is obtained by minimizing the mean square error between the model predicted and measured rates for each video. From Fig. 7, we can conclude (10) can model the bit rate of CMVIA(3D) videos using H.265/HEVC standard with high accuracy.

B. Model Parameter Prediction

In this subsection, we discuss how we adjust the parameter model [(8), (9)] proposed in [23] to cope with CMVIA(3D) application. As is reported in [23] that μ_{FD}, μ_{MVM} and μ_{MVM}/σ_{MDA} are the most related content features which influence parameter α. However, as is shown in Fig. 7, the parameter α will vary for different texture detail settings. Thus we combine texture detail, *TD*, with the content features including σ_{FD}, μ_{MVM}/σ_{MVM}, etc. proposed in [23] as the input parameters for predicting α. Further, we captured 36 30-s-long video clips with different texture detail settings and for different applications performing different tasks. We use the same generalized linear predictor with leave-one-out cross-validation error method reported in [23] to derive and validate the equations, which are shown in (11) and (12)

$$\alpha = B\left[1 \quad \mu_{FD} \quad \mu_{MVM} \quad \frac{\mu_{MVM}}{\sigma_{MDA}} \quad TD\right]^T \tag{11}$$

$$B = \begin{bmatrix} 1.13 & -0.076 & -0.042 & 0.00132 & 0.31 \end{bmatrix} \tag{12}$$

Our results show that using μ_{FD}, μ_{MVM}, μ_{MVM}/σ_{MDA} and *TD* is sufficient to predict the model parameter accurately. Fig. 8 shows the bit rate estimation results comparing the estimated bit rate versus actual bit rate using another 18 30-s video clips encoded with different QPs.

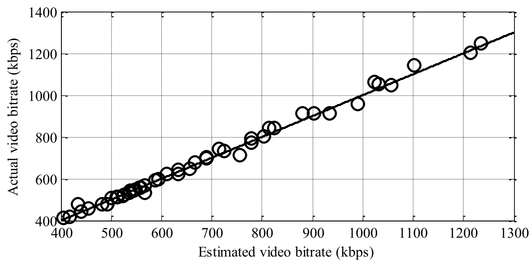


Fig. 8. Validation of bit rate estimation.

Given:

1) Video content features μ_{FD} , μ_{MVM} and σ_{MDA}

2) Network bandwidth limit BW

3) Texture Detail bound TD_{min} and TD_{max}

4) Quantization Parameter bound QP_{min} and QP_{maxx}

Find:

The optimal TD_L, TD_R, QP_L, and QP_R to minimize impairment

$$I^{OPT} = \min I\left(VE, GR\right) = \min I\left(TD_L, TD_R, QP_L, QP_R\right)$$

s.t.

$$TD_{\min} \le TD_L \le TD_R \le TD_{\max}$$

$$QP_{\min} \le QP_L \le QP_R \le QP_{\max}$$

$$R_L\left(TD_L, QP_L\right) + R_R\left(TD_R, QP_R\right) \le BW$$

Fig. 9. Problem formulation.

The correlation is 0.99 indicating the high accuracy of the proposed model.

Thus, (10)– (12) completes our bit rate model.

VI. Optimization Algorithm

In Sections IV and V we have proposed 1) a user experience model which models cloud mobile 3D display gaming user experience as a function of video encoding settings and graphics rendering settings, and 2) a bit rate model which estimates video bit rate needed to encode the rendered video as a function of video encoding settings and graphics rendering settings. In this section, we combine these two models so that by selecting proper graphics rendering (texture detail) settings and video encoding (quantization parameter) settings of the rendered video, we can find an optimal solution for maximizing user experience (minimizing I) given a bandwidth limit.

A. Problem Formulation

Fig. 9 shows the problem formulation. We formulate the problem as an optimization problem. Because according to (1) and (2), maximizing MOS is equal to minimizing I, we set our optimization target as minimizing I. TD_{min}, TD_{max}, QP_{min}, and QP_{max} are the minimum and maximum boundaries of the settings being used for an application.

B. Algorithm Description

We first describe the key ideas and insights of how we analyze the problem and develop the algorithm. Then we discuss the detailed steps of the algorithm.

First, notice that the problem we are to solve contains four variables that are all discrete variables. For a convex or concave problem with continuous variables, it is very easy to solve. However, for the proposed problem, we are faced with two difficulties. 1) It is hard to prove our problem is a convex or concave problem directly or maybe the problem is not convex or concave at all. 2) The variables are discrete. From the literature [1], we can conclude that discrete variable linear programming is very hard to solve, discrete variable convex optimization is even harder and if the problem is not even convex or concave, it will be almost impossible to find a shortcut and prove optimality.

Notice that assuming the variable range space of TD is n, the combination of TD_L and TD_R can only have *2n-1* choices. That is because we only allow either $TD_L = TD_R$ or $TD_L = TD_R - 1$ as in our previous work [1] our subjective test shows when TD_L and TD_R has more than one level difference, it creates a large impairment, so we exclude those cases from our value space. In addition, in reality, n can only be a very small number as it is not possible to define too much texture detail levels. In our experiment, we choose $n = 3$ resulting in five combinations in total that is two orders less than the number of combinations of QP_L and QP_R. Thus, we propose to divide our problem into *2n-1* sub-problems with only two variables, calculate the best solution for each sub-problem and compare to get the final best solution. In this case, one advantage is that the number of variables are decreased so that it becomes possible to prove the sub-problem to be a convex or concave problem. Note that the objective function is a piecewise function; we provide proof in Appendix A to show that one piece of the objective function together with the constraint functions is a convex problem. Thus, we propose to first relax the discrete variable optimization problem into a continuous variable optimization problem. Then we propose to use a relatively small search space that we will discuss in detail later to find the real optimal discrete solution. We can also prove the complexity of the search space is $O(m)$ where m is the possible choices of QP. By utilizing the thoughts and ideas above, we are able to design an algorithm leading to an optimal solution with low complexity.

Fig. 10 shows a block diagram of our proposed algorithm. We use a grey background color to produce a simplified version with Steps a-e and use a white background color to produce a more detailed description with Steps 1–9. We first describe grey blocks. Overall, Step a in grey which is equivalent to Step 1 in white is a pre-process step. It will be reused many times afterwards, so we do it in the beginning to avoid duplicated computation. From Step b to Step c in grey which is equivalent to Step 2 to Step 4 in white, we aim to find the optimal solution for each sub-problem (Note that we may be able to find the discrete optimal solution directly, if not we find a continuous optimal solution). From Step d to Step e in grey which is equivalent to Step 5 to Step 9 in white, we perform two tasks. One task is to find the discrete optimal solutions for those problems with continuous optimal solutions in the previous steps. The other task is to compare the discrete solutions among all sub-problems and find the overall best discrete optimal solution. The diagram looks a little complicated in white especially for Step 5 to Step 9. It is because in order to speed up the process, we change some orders of computation to do some pruning. For example,

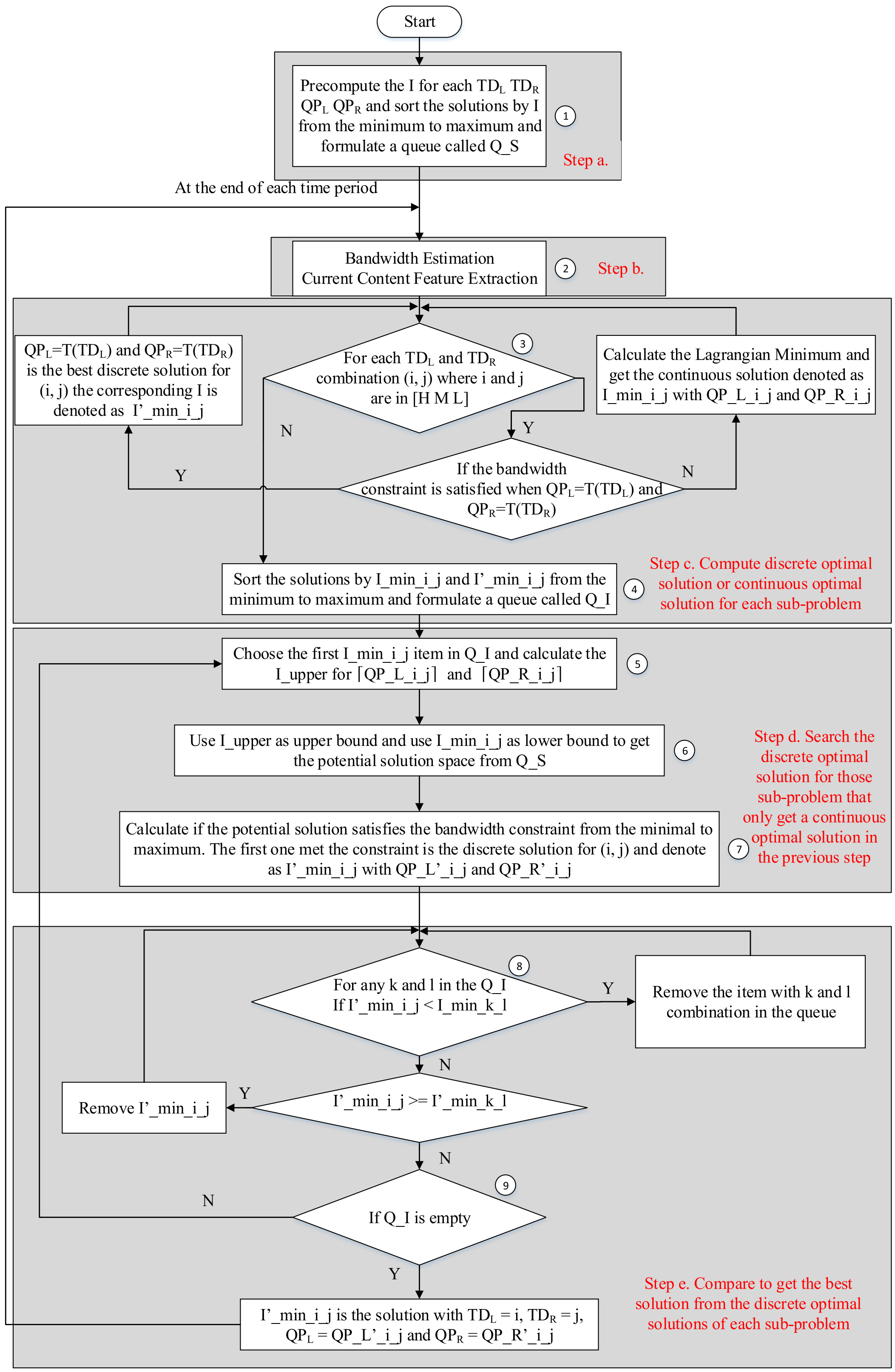


Fig. 10. The block diagram of the algorithm.

if we find the discrete optimal solution for sub-problem 1 is better than the continuous optimal solution for sub-problem 2, then we don't need to compute the discrete optimal solution for sub-problem 2 as it is guaranteed to be worse than the continuous one.

Then we describe white blocks in each grey block. At the beginning, in step 1, we will first compute the values of I for all combinations of the parameters (QP_L, QP_R, TD_L, TD_R) and sort them in a queue called Q_S. As this is a one-time computation and the result can be saved in memory, it saves redundant computations during the execution of the algorithm, and at the same time the sorting itself will be an important step in this algorithm. The rest of the algorithm is designed to be periodically executed according to a fixed time interval. In our experiment, we use 1 s as the interval.

During each interval, in step 2, video content features are extracted to estimate the video bit rate, and network bandwidth is estimated through a network probing method proposed in our previous work [2]. Then, at the end of each time interval, after gathering the necessary information, we divide our problem into *2n-1* sub-problems.

For each sub-problem, in step 3, we first check whether the condition when $QP_L = T(TD_L)$ and $QP_R = T(TD_R)$ satisfies the bandwidth constraint. For asymmetric settings of texture detail, for example $TD_L = M$ and $TD_R = L$, we will also check the conditions when $QP_L > T(TD_L)$ and $QP_R < T(TD_R)$. If these conditions satisfy the bandwidth constraint, the best one with lowest I among them will be the optimal discrete solution for the sub-problem because any other solution [described by $g()$ function in (4)] in this sub-problem will result in higher I. Otherwise, we relax the discrete optimization problem into a corresponding continuous optimization problem and use Lagrangian Minimum method to get the solution for this continuous problem. The proof why the continuous problem is convex and the corresponding equations of how to compute the minimum are provided in Appendix B. After the steps above, for each sub-problem, we can either get a discrete optimal solution or continuous optimal solution. Then, we need to find out the real discrete minimum among all sub-problems. In step 4, we sort these solutions from minimal I to maximal I and put them in a queue called Q_I and then start examining from the minimal continuous solution. The reason why we design Q_I is because we want to sort and compute from the minimal so that if we find a discrete optimal solution that is better than other continuous optimal solutions, we can remove those continuous optimal solutions to achieve the goal of pruning. Notice that for any continuous solution whose corresponding parameters are $QP\_L\_i\_j$ and $QP\_R\_i\_j$ where L means left view, R means right view, i means the choice for TD_L and j means the choice for TD_R, the discrete solution when $QP_L = \lceil QP\_L\_i\_j \rceil$ and $QP_R = \lceil QP\_R\_i\_j \rceil$ will always satisfy the bandwidth constraint because both QPs are increasing, resulting in the bandwidth consumption to be decreasing.

In step 5, we can set this solution as the upper bound *I_upper* and set the continuous solution as the lower bound for this sub-problem. By using this upper bound and lower bound, in step 6, we go back to Q_S and can select the solutions that have lower I than upper bound and higher I than lower bound. In step 7, we compute from the lowest I to the highest I to see if one of those solutions can satisfy the bandwidth constraint and if so the first one that can satisfy the constraint will be the best discrete solution for this sub-problem. Note that in the worst case, the upper bound will be the discrete solution so it is guaranteed to find a solution by this method. It is also proved in Appendix B that the complexity of searching for the discrete solution inside the bounds is $O(m)$ where m is the number of choices of *QP*.

After getting one discrete solution, in step 8, the algorithm will compare it with the continuous or discrete solutions of the other sub-problems. If the current discrete solution is better than the continuous or discrete solutions of the other sub-problems, these solutions in other sub-problems do not need to be further examined to find the corresponding discrete solution. If the current discrete solution is not better than the discrete solutions in other sub-problems, the current solution will be dropped. This pruning will not result in inaccuracy but it will improve the execution time of the algorithm significantly. After the above in step 9, it will judge if Q_I is empty, if so, it can get the final optimal discrete solution for the problem.

C. Complexity Analysis

We define the range of the parameter *TD* as n and that of *QP* as m. Because each parameter can be set in both the left view and the right view, considering a brute force algorithm, the complexity the problem using a brute force algorithm is $O(n^2m^2)$.

In our proposed algorithm (Fig. 10), for each sub-problem, we calculate the continuous optimal and then search for the discrete optimal in a space of complexity m. Thus, the worst case complexity of this algorithm is $O(n^2m)$. Considering the pruning we introduced in the algorithm, that we do not need to search discrete solutions for other sub-problems if the first discrete solution is better than all the other continuous solutions then, the complexity of the algorithm can be reduced to $O(n^2+m)$. In the experiments we conducted and reported in Section VII, we see that the algorithm prunes successfully in more than 90% of the cases.

VII. Experimental Results

In this section, we report on experiments conducted using a commercial cloud service, Amazon Web Service (AWS) [28], to verify the performance improvement by applying the proposed *JAVRE* technique. We use the same testbed as shown in Fig. 2, except that 1) we put a network emulator Linktropy [29] between AP and laptop to control wireless network condition, and 2) we implement our CMVIA(3D) system, including the *JAVRE* algorithm, on AWS servers. In detail, we modified the open-source game engine Plane shift and open-source virtual world application SecondLife, so that 1) the rendering engine is able to pre-load different levels of textures when initializing the rendering loop and 2) when one specific texture detail setting is chosen it is able to switch to it dynamically. We also programmed a control software that is able to 1) capture the game scene or virtual world scene from the cloud server screen buffer in real time, 2) encode the left and right view videos by x265 library, 3) stream the videos to the client devices through TCP, 4) probe the network and estimate the available network bandwidth, and 5) with the *JAVRE* algorithm implemented, decide the parameters *TD* s and *QP* s that result in best use experience and set them in the rendering engine through process level share memory mechanism. For the Amazon cloud server, the CPU is Intel Xeon E5-2670 @2.60 GHz with 15 GB memory and the

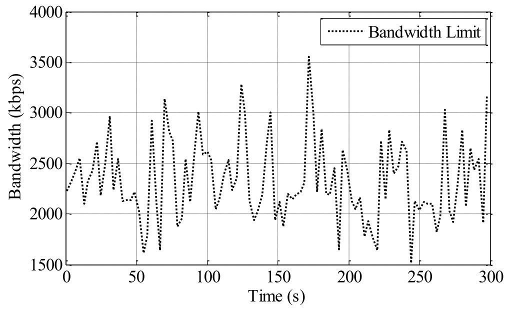


Fig. 11. LTE bandwidth trace.

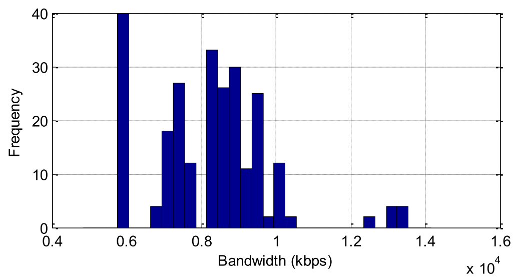


Fig. 12. PDF of the bandwidth from Amazon cloud server to UCSD mobile systems design lab.

GPU has 1536 CUDA cores and 4 GB of video memory. The operating system is Windows Server 2008 R2 SP1.

We firstly collected real 4G-LTE network traces by using network bandwidth testing software Speedtest.net [30] to record the bandwidth. Fig. 11 shows a sample LTE trace, which is emulated using the network emulator in our test-bed. We then measured the bandwidth from Amazon cloud server to our lab. We use iPerf software to test, and collect bandwidth value every 10 min from 8:00 a.m. to 10:00 p.m. for three days. Fig. 12 shows the PDF of the results. We find that the bandwidth has some variance but is quite adequate. Compared to Fig. 11 where the lowest bandwidth of LTE is about 3.4 Mb/s, the largest bandwidth from AWS to our lab is about 5.8 Mb/s. Thus we conclude the bandwidth bottleneck on the entire transmission flow is caused by the LTE trace. In addition, for comparison reasons, we also implemented two other algorithms called *ARA* from [2] and *JREA* from [6]. Basically, *ARA* enables the game to set two different texture details for the left view and the right view, but video encoding settings are fixed to the highest values. *ARA* also enables view distance settings which let the game not render the objects whose distance to the virtual camera is greater than a certain threshold (view distance threshold). *JREA* is an adaptation algorithm developed for 2D CMG applications. The basic idea for this technique is that it pre-defines several groups of parameter combinations and assigns them into different levels. The algorithm chooses to go up a level or go down a level at a time when the network conditions changes. We extended the framework of it for CMVIA(3D), and evaluated the performance of the algorithms on all three applications (gaming, virtual classroom and virtual art gallery).

Fig. 13(a)–(c), Fig. 14(a)–(c), and Fig. 15(a)–(c) show the results for the three different cloud 3D virtual applications, cloud mobile 3D display gaming, cloud mobile 3D virtual classroom and cloud mobile 3D virtual art gallery, respectively. Fig. 13(a), Fig. 14(a), and Fig. 15(a) plot the video bit rate while Fig. 13(b), Fig. 14(b), and Fig. 15(b) plot the corresponding impairment I and Fig. 13(c), Fig. 14(c), and Fig. 15(c) show the *MOS* scores obtained. In each figure, we compare the performance of the three algorithms (*JAVRE*, *ARA* and *JREA*). The average values of I and *MOS* for each algorithm and each application are also shown in Table VIII. From the figures and the table, we can make the following observations:

TABLE VIII
STATISTICAL RESULTS OF THE EXPERIMENT SHOWING IMPAIRMENT I AND OVERALL USER EXPERIENCE MOS

Application	I			MOS		
	JAVRE	ARA	JREA	JAVRE	ARA	JREA
Gaming	13.93	42.44	57.95	4.13	3.10	2.51
Virtual Classroom	7.64	26.59	40.31	4.35	3.71	3.16
Virtual Art Gallery	17.70	44.31	60.81	3.67	2.92	2.11

1) In all applications, *JAVRE* performs the best (result in lowest I and highest *MOS*), *ARA* is the next and *JREA* is the worst. The improvement in terms of *MOS* score by using *JAVRE* over *ARA* is up to 33.2% and that over *JREA* is up to 64.5%.
2) For all the three algorithms, the *MOS* score of gaming are all lower than that of virtual classroom. The reason is that a) the parameters of the user experience model for gaming and virtual classroom are very similar and b) for virtual classroom application, the virtual camera is mostly fixed with limited movement, unlike in gaming where the camera is mostly moving. Hence the virtual classroom 3D video streamed from the cloud has lower video bit rate for the same parameter setting, and therefore under the same bandwidth constraint, it can choose *TD* s and *QP* s to result in lower I and higher *MOS*, than is possible for the 3D gaming application.
3) For all the three algorithms, the *MOS* values of virtual art gallery are all lower than that of virtual classroom. The reason is in virtual art gallery application, the viewers pay extensive attention to the details of the virtual paintings and hence will be more sensitive to the influence of decreasing *TD* or increasing *QP*. Therefore, although for the virtual art gallery application, the algorithm will choose parameters resulting in more bandwidth consumption, but it still does not provide the user experience as well as it does for the virtual classroom application.

The results show that there can be variations in user experiences achieved depending on the application, but they 1) validate the feasibility of our proposed CMVIA(3D) architecture to enable wireless 3D virtual immersive user experiences with the applications running on the cloud, and 2) show the effectiveness of the proposed approach (*JAVRE*) to enable high quality 3D user experiences for a wide variety of applications, from 3D gaming to 3D virtual reality applications like virtual classroom and virtual art gallery.

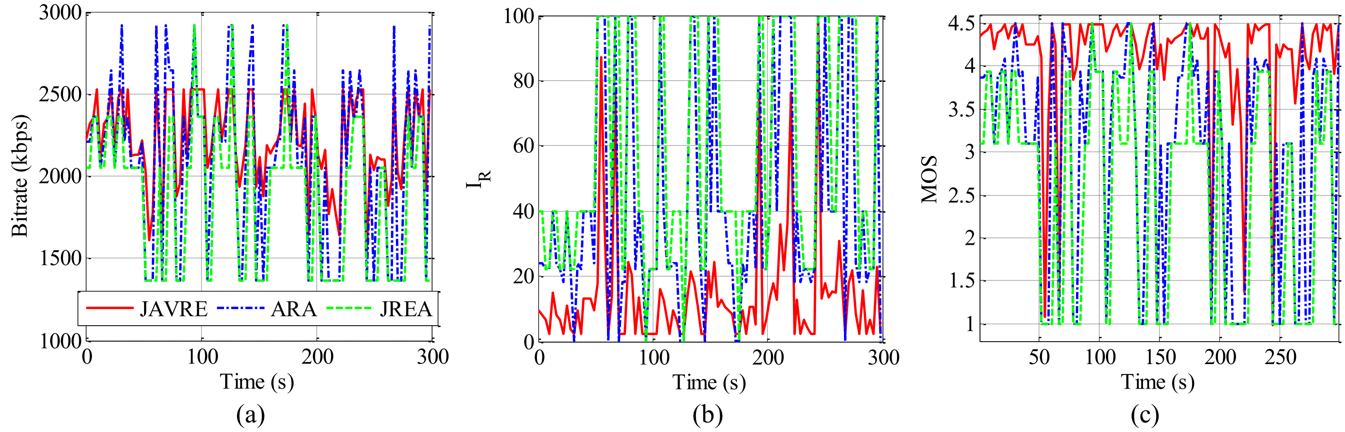


Fig. 13. (a) left, bandwidth consumption (b) middle, resulting impairment *I* (c) right, *MOS*score obtained, using *JAVRE*, *ARA*, and *JREA* for cloud mobile 3D gaming application.

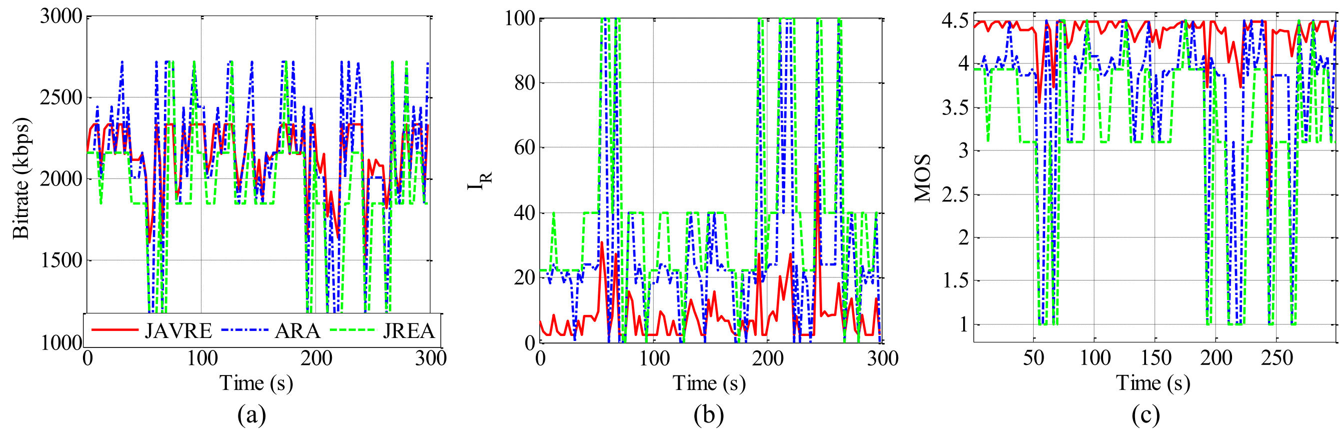


Fig. 14. (a) left, bandwidth consumption (b) middle, resulting impairment *I* (c) right, *MOS* score obtained, using *JAVRE*, *ARA*, and *JREA* for cloud mobile 3D virtual classroom application.

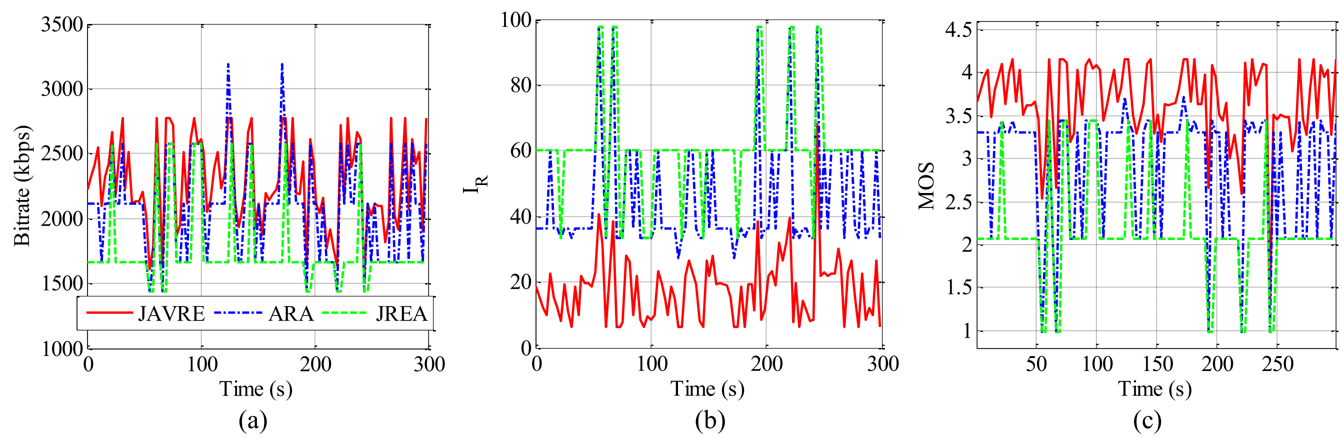


Fig. 15. (a) left, bandwidth consumption (b) middle, resulting impairment *I* (c) right, *MOS* score obtained, using *JAVRE*, *ARA*, and *JREA* for cloud mobile 3D virtual art gallery application.

VIII. Conclusion

The main contributions of this paper are the following.

1) We performed extensive subjective tests to derive a general user experience model for cloud mobile 3D virtual immersive applications considering both asymmetric video encoding and graphics rendering.

2) We derived a bit rate model which is suitable for H.265/HEVC standard, by taking into account both video encoding parameters and graphics rendering parameters.

3) We developed a novel adaptation algorithm called *JAVRE* that can decide *QP* s and *TD* s dynamically by making use of the above two models to ensure best user experience for

cloud mobile 3D virtual immersive applications under dynamic wireless network conditions.

4) By conducting experiments using real 4G-LTE network profiles on commercial cloud service with three different virtual immersive applications, we demonstrated significant improvement over existing methods in user experience when the proposed *JAVRE* algorithm is applied.

In the future, because the technique proposed in this paper can be independent of the encoding technology that means the improvement from upgrading encoder will be additive on top of the gains made by our proposed technique, we plan to explore the use of more advanced codes such as HEVC-SCC [19], MV-HEVC [32], and 3D-HEVC [33] in performing joint asymmetric rendering and encoding adaptation to further optimize the delivery of cloud based 3D virtual immersive applications. In addition, we are very interested in investigating how to combine more video encoding parameters such as framerate and resolution with graphics rendering parameters to further optimize the system. Finally, we also plan to perform further subjective tests to model and subsequently optimize the user experience for virtual reality applications with Head Mounted Displays (HMD) such as Oculus Rift [34], HTC Vive [35], PlayStation VR [36], Samsung Gear VR [37], etc.

APPENDIX

A. Proof of Convexity

In the following, we will first prove that for each sub-problem, when $g()$ function applies, the problem is a convex problem and when it relaxes into a continuous problem, it can be solved by Lagrangian Minimum method. We will also derive the equations for the minimal solution.

According to [27], an optimization problem of the form

$$\begin{aligned} &min\ f(x) \\ &s.t.\ h_i(x) \le 0, \quad i = 1, \ldots, n \end{aligned} \tag{13}$$

is called convex if the functions $f, h_1, \ldots, h_m : R^n \to R$ are convex [27].

Recall that the objective function (6) when $g()$ function applies is

$$g(x_1, x_2) = a_1 x_1 + a_2 x_2 + b_1 \tag{14}$$

where $x_1 = QP_L - T(TD_L)$ and $x_2 = QP_R - T(TD_R)$

For a given sub-problem, TD_L and TD_R are fixed. Thus

$$g(QP_L, QP_R) = c_1 QP_L + c_2 QP_R + c_3 \tag{15}$$

Since the $g()$ function is a bilinear function, it can be considered as both a convex function and a concave function.

In addition, for a given sub-problem, TD_L and TD_R are fixed, the corresponding constraint function is

$$R(QP_L, QP_R) = R(QP_L) + R(QP_R) \tag{16}$$

If we combine all the constants together, the constraint function can be rewritten as

$$R(QP_L, QP_R) = e_1 2^{d_1(QP_L-4)} + e_2 2^{d_2(QP_R-4)} \tag{17}$$

Because the Hessian Matrix of the above equation is

$$H = \begin{bmatrix} (d_1 \ln 2)^2 e_1 2^{d_1(QP_L-4)} & 0 \\ 0 & (d_2 \ln 2)^2 e_2 2^{d_2(QP_R-4)} \end{bmatrix} \tag{18}$$

And also because $e_1 > 0$ and $e_2 > 0$, H is a positive definite matrix. Thus, the constraint function is a convex function.

The range of QP s are from 0 to 51 and we also have $QP_L <= QP_R$. Thus, the range of the variable is a closed convex set.

Therefore, our problem is a convex optimization problem.

After proving that the problem is a convex optimization problem, we can compute the minimal value by the following method.

First, construct an $F()$ function as

$$F(QP_L, QP_R) = g(QP_L, QP_R) + \lambda(R(QP_L, QP_R) - BW) \tag{19}$$

Let

$$\begin{cases} \dfrac{\partial g(QP_L, QP_R)}{\partial QP_L} + \lambda \dfrac{\partial R(QP_L, QP_R)}{\partial QP_L} = 0 \\ [2pt] \dfrac{\partial g(QP_L, QP_R)}{\partial QP_R} + \lambda \dfrac{\partial R(QP_L, QP_R)}{\partial QP_R} = 0 \\ R(QP_L, QP_R) = BW \end{cases} \tag{20}$$

Hence, we can solve QP_L and QP_R as

$$\begin{aligned} QP_L &= \frac{\log_2\left(\frac{BW}{e_1} \cdot \frac{c_1}{c_1+c_2}\right)}{d_1} + 4 \\ QP_R &= \frac{\log_2\left(\frac{BW}{e_1} \cdot \frac{c_2}{c_1+c_2}\right)}{d_1} + 4 \end{aligned} \tag{21}$$

Thus, we prove that the problem is convex and also provide the equations for the optimal solution.

B. Proof of Complexity

In the following, we will prove the search space of the best discrete solution of a sub-problem is of the complexity m where m is the variable range of QP.

We first generalize the problem and prove the more general problem, so that our problem is just a special case under this framework. We can state the generalized problem below.

Define:

$$g(x, y) = ax + by + c$$

Assume:

1. Both $a > 0$ and $b > 0$
2. x and y are both integers and both ranges in *(0 t]*.

Consider:

$$\begin{aligned} &\forall\ (x, y) \\ &\exists k\ pairs\ of\ (x', y')\ in\ total \\ &s.t.\ f(x, y) \le f(x', y') \le f(x+1, y+1) \end{aligned}$$

To prove:

$k \le mt$, where m is a constant

We provide proof to this problem in the following:

Proof:

$$f(x, y) \leq f(x', y') \leq f(x+1, y+1)$$
$$\Leftrightarrow ax + by \leq ax' + by' \leq ax + by + a + b$$
$$\Leftrightarrow x + \frac{b}{a}y \leq x' + \frac{b}{a}y' \leq x + \frac{b}{a}y + 1 + \frac{b}{a}$$
$$Let\ \Delta x = x' - x,\ \Delta y = y' - y$$
$$\Leftrightarrow 0 \leq \Delta x + \frac{b}{a}\Delta y \leq 1 + \frac{b}{a}$$
$$\Leftrightarrow -\frac{a}{b}\Delta x \leq \Delta y \leq 1 + \frac{a}{b} - \frac{a}{b}\Delta x$$
$$\therefore \text{For any } \Delta x,$$
$$\Delta y \text{ can have at most } 1 + \frac{a}{b} \text{ values}$$
$$\because x' = x + \Delta x \in (0\ t],\ y' = y + \Delta y \in (0\ t]$$
$$\therefore \Delta x \in [-x\ t - x],\ \Delta y \in [-x\ t - x]$$
$$\because \text{Both } x \text{ and } x\text{' are integers}$$
$$\therefore \Delta x \text{ is also an integer}$$
$$\therefore \Delta x \text{ can have } t \text{ values}$$
$$\therefore \Delta y \text{ can have at most } t\left(1 + \frac{a}{b}\right) \text{ values}$$
$$\therefore \mathbf{k} \leq t\left(1 + \frac{a}{b}\right), \text{ where } 1 + \frac{a}{b} \text{ is a constant}$$

REFERENCES

[1] Y. Lu, Y. Liu, and S. Dey, "Enhancing cloud mobile 3D display gaming user experience by asymmetric graphics rendering," in *Proc. IEEE Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2014, pp. 368–374.

[2] Y. Lu, Y. Liu, and S. Dey, "Modeling and optimizing cloud mobile 3D display gaming user experience by asymmetric graphics rendering," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 3, pp. 1–16, Apr. 2015.

[3] C.-Y. Huang, K.-T. Chen, D.-Y. Chen, H.-J. Hsu, and C.-H. Hsu, "GamingAnywhere: The first open source cloud gaming system," *ACM Trans. Multimedia Comput., Commun. Appl.*, vol. 10, no. 1s, Jan. 2014, Art. no. 10.

[4] W. Cai, V. C. M. Leung, and M. Chen, "Next generation mobile cloud gaming," in *Proc. IEEE 7th Int. Symp. Service Oriented Syst. Eng. (SOSE)*, Mar. 2013, pp. 551–560.

[5] R. Shea, J. Liu, E. C.-H. Ngai, and Y. Cui, "Cloud gaming: Architecture and performance," *IEEE Netw.*, vol. 27, no. 4, pp. 16–21, Jul./Aug. 2013.

[6] S. Wang and S. Dey, "Adaptive mobile cloud computing to enable rich mobile multimedia applications," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 870–883, Jun. 2013.

[7] L. R. Porter, *Creating the Virtual Classroom: Distance Learning With the Internet*. New York, NY, USA: Wiley, 1997.

[8] S. Hrk, "Virtual art gallery," in *Proc. 5th Central Eur. Seminar Comput. Graph.*, 2001, pp. 185–194.

[9] H. Afsarmanesh and L. M. Camarinha-Matos, "Future smart-organizations: A virtual tourism enterprise," in *Proc. IEEE 1st Int. Conf. Web Inf. Syst. Eng.*, vol. 1, 2000, pp. 456–461.

[10] G. Saygili, C. G. Gurler, and A. M. Tekalp, "Evaluation of asymmetric stereo video coding and rate scaling for adaptive 3D video streaming," *IEEE Trans. Broadcast.*, vol. 57, no. 2, pp. 593–601, Jun. 2011.

[11] S. Valizadeh, M. Azimi, and P. Nasiopoulos, "Bitrate reduction in asymmetric stereoscopic video with low-pass filtered slices," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2012, pp. 170–171.

[12] Video Codec for Audiovisual Services at p x 64 Kbit/s, Version 1, document ITU-T Rec. H.261, Nov. 1990.

[13] Video Coding for Low Bit Rate Communication, document ITU-T Rec. H.263, Nov. 1995.

[14] Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 Mbit/s-Part 2: Video, document ISO/IEC 11172-2 (MPEG-1), ISO/IEC JTC 1, 1993.

[15] Coding of Audio-Visual Objects-Part 2: Visual, document ISO/IEC 14496-2 (MPEG-4 Vis. Version 1), ISO/IEC JTC 1, Apr. 1999.

[16] Generic Coding of Moving Pictures and Associated Audio Information-Part 2: Video, document ITU-T Rec. H.262 ISO/IEC 13818-2 (MPEG 2 Video), ITU-T ISO/IEC JTC 1, Nov. 1994.

[17] Advanced Video Coding for Generic Audio-Visual Services, document ITU-T Rec. H.264 ISO/IEC 14496-10 (AVC), ITU-T ISO/IEC JTC 1, May 2003.

[18] High Efficiency Video Coding, document ITU-T Rec. H.265 and ISO/IEC 23008-2 (HEVC), ITU-T and ISO/IEC JTC, Jan. 2013.

[19] J. Xu, R. Joshi, and R. A. Cohen, "Overview of the emerging HEVC screen content coding extension," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 50–62, Jan. 2016.

[20] x265 Software. [Online]. Available: http://x265.org/

[21] B. Vankeirsbilck *et al.*, "Quality of experience driven control of interactive media stream parameters," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2013, pp. 1282–1287.

[22] Y. Liu, S. Wang, and S. Dey, "Content-aware modeling and enhancing user experience in cloud mobile rendering and streaming," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 4, no. 1, pp. 43–56, Mar. 2014.

[23] Z. Ma, M. Xu, Y.-F. Ou, and Y. Wang, "Modeling of rate and perceptual quality of compressed video as functions of frame rate and quantization stepsize and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 5, pp. 671–682, May 2012.

[24] B. Lee and M. Kim, "Modeling rates and distortions based on a mixture of Laplacian distributions for inter-predicted residues in quadtree coding of HEVC," *IEEE Signal Process. Lett.*, vol. 18, no. 10, pp. 571–574, Oct. 2011.

[25] Planeshift. [Online]. Available: http://www.planeshift.it/

[26] SecondLife. [Online]. Available: http:// www.secondlife.com/

[27] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[28] Amazon Web Service. [Online]. Available: http://aws.amazon.com

[29] Linktropy. [Online]. Available: http://www.apposite-tech.com/products/

[30] Speedtest.net. [Online]. Available: https://itunes.apple.com/tw/app/speedtest.net-speed-test/id300704847?mt=8

[31] S. Dey, Y. Liu, S. Wang, and Y. Lu, "Addressing response time of cloud-based mobile applications," in *Proc. ACM 1st Int. Workshop Mobile Cloud Comput. Netw.*, Jul. 2013, pp. 3–10.

[32] G. Tech, Y. Chen, K. Müller, J.-R. Ohm, A. Vetro, and Y.-K. Wang, "Overview of the multiview and 3D extensions of high efficiency video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 35–49, Jan. 2016.

[33] G. J. Sullivan, J. M. Boyce, Y. Chen, J.-R. Ohm, C. A. Segall, and A. Vetro, "Standardized extensions of high efficiency video coding (HEVC)," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 1001–1006, Dec. 2013.

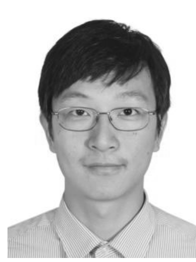
[34] Oculus Rift. [Online]. Available: https://www.oculus.com/

[35] HTC Vive. [Online]. Available: https://www.htcvive.com/

[36] PlayStation VR. [Online]. Available: https://www.playstation.com/en-au/explore/ps4/features/playstation-vr/

[37] Samsung Gear VR. [Online]. Available: http://www.samsung.com/us/explore/gear-vr/

[38] *The E-Model: A Computational Model for Use in Transmission Planning*, document ITU-T Rec. G.107, Mar. 2005.



Yao Lu (S'13) currently a Ph.D. degree student at University of California, San Diego, La Jolla, CA, USA.

His research interests include mobile multimedia, computer graphics, video encoding, computer networks, and cloud computing.

Sujit Dey (SM'03–F'14) received the Ph.D. degree in computer science from Duke University, Durham, NC, USA, in 1991.

He is a Professor in the Department of Electrical and Computer Engineering, University of California, San Diego (UCSD), La Jolla, CA, USA, where he heads the Mobile Systems Design Laboratory, which is developing innovative technologies in mobile cloud computing, adaptive multimedia and networking, green computing and communications, and predictive and prescriptive analytics to enable future applications in connected health, immersive multimedia, smart cities, and smart factories. He is the Director of the Center for Wireless Communications, and the Director of the newly launched Institute for the Global Entrepreneur at UCSD. He served as the Faculty Director of the von Liebig Entrepreneurism Center from 2013–2015, and as the Chief Scientist, Mobile Networks, at Allot Communications from 2012–2013. He founded Ortiva Wireless in 2004, where he served as its founding CEO and later as CTO and Chief Technologist until its acquisition by Allot Communications in 2012. Prior to Ortiva, he served as the Chair of the Advisory Board of Zyray Wireless till its acquisition by Broadcom in 2004, and as an advisor to multiple companies including ST Microelectronics and NEC. Prior to joining UCSD in 1997, he was a Senior Research Staff Member at NEC C&C Research Laboratories in Princeton, NJ, USA. He has co-authored more than 200 publications, and a book on low-power design. He holds 18 U.S. and two international patents, resulting in multiple technology licensing and commercialization.

Dr. Dey has been a recipient of six IEEE/ACM Best Paper Awards, and has chaired multiple IEEE conferences and workshops.