ISSN 1047-3203
Volume 21, Issues 5–6, July/August 2010
ELSEVIER
JOURNAL OF
VISUAL
Communication &
IMAGE
Representation
Editors-in-Chief
Yehoshua Y. Zeevi
C.-C. Jay Kuo
Available online at www.sciencedirect.com
ScienceDirect

J. Vis. Commun. Image R. 21 (2010) 523–532

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

RD-optimized interactive streaming of multiview video with multiple encodings

Yanwei Liu [a,b,*], Qingming Huang [a,b], Siwei Ma [c], Debin Zhao [d], Wen Gao [c]

[a] *Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China*
[b] *Graduate School of Chinese Academy of Sciences, Beijing, China*
[c] *Institute of Digital Media, Peking University, Beijing, China*
[d] *Department of Computer Science, Harbin Institute of Technology, Harbin, China*

ARTICLE INFO

Article history:
Received 29 July 2009
Accepted 30 January 2010
Available online 15 February 2010

Keywords:
Interactive streaming
Multiview video
Multiple encodings
Rate-distortion
H.264/MVC
RD-optimized scheduling
End-to-end distortion model
Error-resilience

ABSTRACT

This paper presents a rate-distortion (RD) optimized interactive streaming method for multiview video pre-compressed by H.264 Joint Multiview Video Model (JMVM). In the proposed method, multiple encodings are first used to facilitate the flexible server–client interaction. Second, a RD-optimized scheduling strategy is provided to guarantee the optimal view-dependent delivery of multiview video. In the RD-optimized scheduling strategy, a distortion model is proposed to estimate the expected end-to-end distortion by accounting for both coding and packet-loss-induced distortions, as well as rendering-induced distortion. With the end-to-end distortion model, the server can select the optimal encoding combination for transmission. Experimental results demonstrate that the proposed method can achieve a significant end-to-end RD performance improvement over the selective streaming methods with simulcast coding or scalable multiview coding. In addition, it has better error-resilience performance to combat with packet-losses over the Internet protocol (IP) networks.

1. Introduction

With the fast advancement in imaging and display technologies, multiview video becomes increasingly popular due to its exciting viewing experiences. Multiview video captures the real scene from different viewing positions and viewing angles, and further provides the users with exciting visual effects. Besides the interactive browsing of visual scene, multiview video brings the end users 3D impressions because of the different scene depth information recorded by multiple cameras. With these characteristics, multiview video is widely used as a kind of new media service in 3D-TV and free viewpoint video systems [1,2].

Interactive multiview video is designed to provide multiview video services with the high degree of interactivity over the Internet [3]. In general, there are two ways to implement the interactive feature for such multiview video systems with patterns of server–clients. The first one is all video streams are sent to a client. The client has enough visual stuff for selection. For this kind of transmission, a lot of channel bandwidth resources are wasted because a client does not need to watch all of the views at one time instant. The second method is that the requested video streams are delivered to a client on demand with an interactive mode. Because only the requested video streams are transmitted over the network at the same time, this method can save the bandwidth effectively with the expense of a little delay compared with the first method.

Due to the channel bandwidth's constraint, multiview video must be compressed before transmission. There are usually two kinds of methods to compress the multiview video. One is simulcast coding in which all views are coded independently. Another is jointly coding of multiview video in which the inter-view correlation between adjacent views is further exploited. Many joint coding algorithms have been proposed to improve the coding efficiency [4–9]. Among these algorithms, most of them make use of multiple reference structure to support inter-view prediction for efficient compression and thus introduce more interdependency between frames. This kind of joint multiview video coding (MVC) is very efficient for auto-stereoscopic display applications, such as 3D-TV with multiple-projectors display [5], which needs all views to be transmitted to the client in a broadcasting way. However, it is not very efficient for interactive delivery of multiview video because the inter-view prediction limits the random accessibility of each frame [10,11]. Although simulcast coding provides the interactive flexibility of view switching and is propitious to interactive delivery of video, it has the lower compression efficiency than the jointly multiview coding, and correspondingly requires more transmission bandwidth.

Recently, some end-to-end streaming methods have been investigated to transmit the multiview video [12]. However, most of them focus on the 3D-TV application. Few works are aiming at

[*] Corresponding author. Address: Institute of Computing Technology, Chinese Academy of Science, Beijing, China.
E-mail address: ywliu@jdl.ac.cn (Y. Liu).

1047-3203/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2010.02.004

the interactive multiview video applications with joint MVC, such as the emerging H.264/MVC standard. In recent years, there are also some related works on interactive streaming, which only transmit the desired contents when they are required at the client's end. The transmission of region of interest (ROI) for high resolution video or image is a typical application [13,14]. The streaming of concentric mosaic [15] and the delivery of light field are also the other applications of interactive streaming [16].

To interactively transmit multiview video over IP channels, many multicast streaming methods are proposed to deliver the video data compressed by simulcast coding of each view [17,18]. For jointly coded multiview video streaming, which delivers the jointly coded multiview video, E. Kurutepe et al. [19] first proposed a scalable multiview video coding (SMVC) scheme to support client-driven selective streaming for interactive 3D-TV application. The proposed system can provide flexibility in bandwidth allocation to the selected views. However, this scenario is not suitable for the error-prone channel. Once packet loss occurs in the base-layer, the overall distortion will deteriorate because the enhancement layer has a strong dependency on the base layer.

This paper proposes a multiview video streaming framework which interactively delivers the multiple video streams compressed by jointly multiview coding. First, we consider a multiple encodings method to guarantee the interactive delivery of multiview video with flexible viewpoint switching. Second, multiview video streams are RD optimally assembled according to the user's request. Such dynamic selective transmission can maximize the quality of multiview video services over error-prone channels. In the RD-optimized scheduling policy, the rendering-induced distortion as well as coding and packet-loss-induced distortions is considered for the end-to-end distortion estimation.

The rest of the paper is organized as follows. Interactive multiview streaming architecture with multiple encodings is illustrated in Section 2. In Section 3, we propose the RD-optimized streaming of multiple encodings over IP networks. In this section we will explain how to estimate the end-to-end distortion and then find the RD optimal encoding combinations. Section 4 presents the experimental results. Finally, some concluding remarks are provided in Section 5.

2. Interactive multiview streaming architecture

Multiview video services, which supply 2D or 3D special visual effects, are deployed on the Internet by a kind of sender–receivers streaming system. In this paper, we consider a unicast streaming method for interactive multiview services. Fig. 1 is the architecture of the interactive multiview streaming system. For the error-prone network, I. Radulovic et al. [20] utilized multiple versions coding for adaptive streaming. Here, in order to facilitate the fast viewpoint switching, we extend the multiple versions concept to the interactive multiview streaming. The notation of multiple encodings is proposed to explain the different coding versions for the same view with different inter-view structures. To achieve the trade-off between the bandwidth consumption and the low-delay viewpoint switching, the multiview videos are offline encoded with multiple encodings. The corresponding RD trace data are also saved in the server.

In order to transmit the data set that the user wants to see at a particular instant, the server communicates with the receiver in an interactive way. The receiver first requests the demanding views, and the viewing position information of a client is sent to the server through a feedback channel. Then, after getting the requesting information and packet loss information, the scheduler extracts the most appropriate data sets via the specific scheduling policy. At last, the server responds the receiver with the requested views data. At the receiver side, when the user's browsing-viewpoint is not located at the original capturing view position, the virtual view will be synthesized by the rendering module to achieve smooth viewpoint transition in free viewpoint browsing.

Multiple encodings encode the multiview videos with several representations. Fig. 2 describes the multiple encodings structure. Each encoding is a coding version of the current view with corresponding inter-view prediction structure which embodies a certain RD performance. To reduce the view switching delay, the spatial down-sampled encodings with low quality are also encoded. The spatial down-sampled encodings assist to switch the current viewpoint to the low-quality encodings when the requested viewpoint does not arrive in time [21]. In the proposed unicast interactive streaming scenario, it generally involves the consideration of viewing delay. Currently, the total delay bound is set to 300 ms. Starting from the user's request, if the responding delay time surpasses 300 ms, the last viewing picture will be repeatedly displayed or the down-sampled encodings of adjacent views will be displayed if the down-sampled encodings are available. Once the request is satisfied within the total delay constraint, the viewing picture will be updated.

Multiple encodings can provide the multiview system with the RD-optimized viewpoint switching capability. The more encodings the system administrates, the better switching capability the system can achieve. However, more encodings will bring more com-

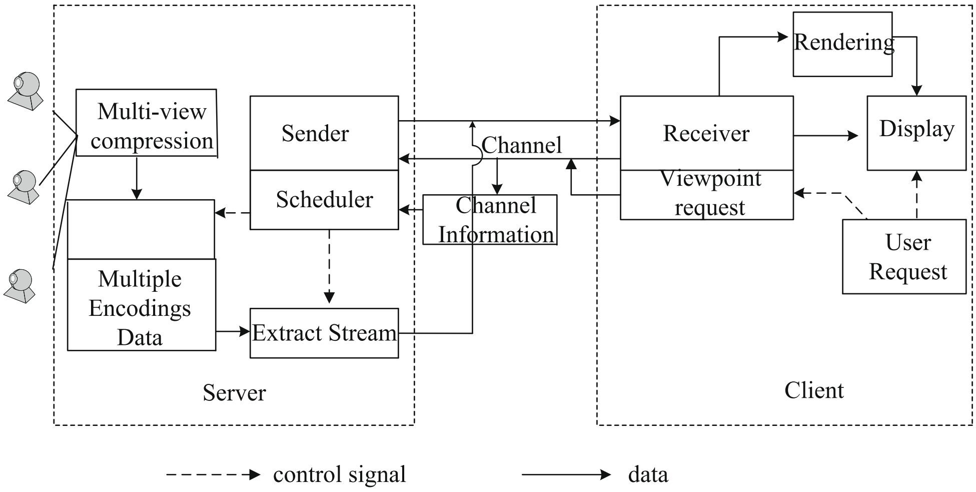


Fig. 1. The architecture of interactive multiview streaming.

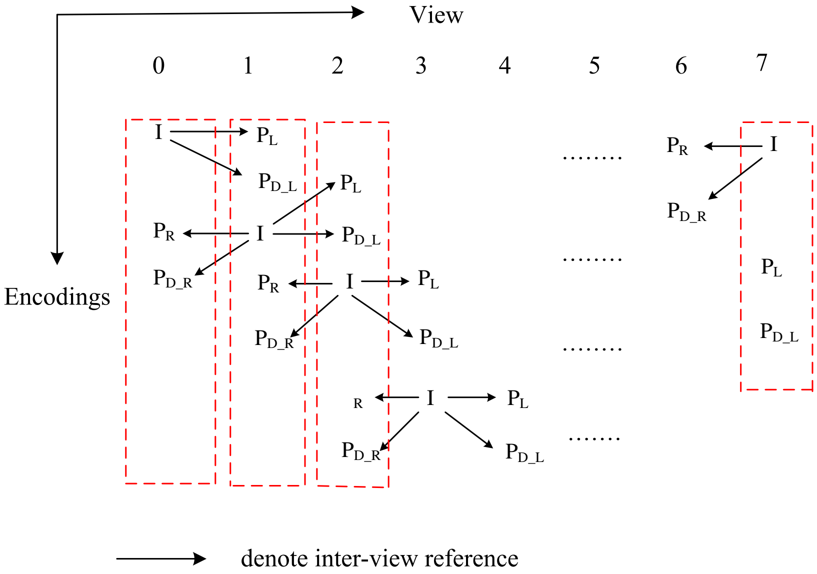


Fig. 2. Multiple encodings structure.

putational complexity. The optimum encoding number can be decided according to the system's requirement of view-switching flexibility. In Fig. 2, each view sequence is encoded by three or five encodings. I denotes that the current view is encoded only with temporal prediction. Assume I_L and I_R denote I encodings of left and right reference views of the current view. P_{D_L} and P_{D_R} denote that the current down-sampled view is encoded with inter-view prediction from I_L and I_R, respectively. P_L and P_R denote the current view predicted from I_L and I_R, respectively. The specific definitions of P_{D_L}, P_{D_R}, P_L and P_R encodings are shown in Fig. 3 and Fig. 4 shows the coding structure of one Group of GOP (GoGOP) with two views. Hierarchical B pictures with GOP size of four frames are used for low-delay random accessibility.

3. RD-optimized multiview streaming

An efficient interactive multiple encodings streaming system should solve two critical issues: the first is how the server should allocate the available resources to guarantee the end-to-end quality of services over error-prone networks; the second is how the server should select and transmit the appropriate encoding combinations of views to maximize the rendering quality of the requested virtual view.

To maximize the streaming performance, the server scheduler manages the dynamic transmission of the multiple encodings data through a RD-optimized scheduling policy. With the pre-saved RD information, the server estimates the end-to-end distortion given the instantaneous feedback of packet loss probability, and then predicts the RD costs of different encoding combinations for one GoGOP using a Lagrangian R–D function:

$$J(\lambda) = E\{D_{end\text{-}to\text{-}end}\} + \lambda R, \tag{1}$$

where $E\{D_{end\text{-}to\text{-}end}\}$ is the expected distortion of one GoGOP and R is the corresponding rate. After achieving the RD cost, the server scheduler selects the most appropriate data unit dynamically to adapt to the user's request. In the current streaming architecture, the whole GoGOP is taken as one data unit. The server responds the user's request once every GoGOP.

3.1. End-to-end distortion model

When the requested view is the captured view, the end-to-end visual quality is only associated with quantization coding and packet loss. Let D_{coding} indicate the quantized distortion and $E\{D_{packet\_loss}\}$ denote the expected packet-loss-induced distortion. According to an additive distortion model [22], the expected end-to-end distortion is then expressed as

$$E\{D_{end\text{-}to\text{-}end}\} = D_{coding} + E\{D_{packet\_loss}\}. \tag{2}$$

When the requested view locates at the middle position between two captured views, the requested view needs to be rendered using adjacent source views. Since the rendering quality is not very perfect due to the blocking artifacts, post-filtering is usually utilized to promote the rendering quality. This operation incurs some visual changes, which can be treated as another kind of distortion. In [23], the rendering view distortion is regarded as a linear combination of the coding distortions of different views when no packet loss occurs. Once the network is error-prone, the rendering view distortion should be a linear combination of the sum of coding and packet loss induced distortions. Since the post-filtering induced distortion is also an additive item for the total distortion for a rendered view, as illustrated in (11) of Section 3.1.2, the end-to-end distortion for a rendered view can be an additive form

$$\begin{aligned} E\{D_{end\text{-}to\text{-}end}\} = {} & D_{coding\_transfered} + E\{D_{packet\_loss\_transfered}\} \\ & + E\{D_{rendering}\}, \end{aligned} \tag{3}$$

where $D_{coding\_transfered}$ and $E\{D_{packet\_loss\_transfered}\}$ represent the coding distortion and expected transmission distortion transferred from the sources reference views, respectively. $E\{D_{rendering}\}$ denotes the expected rendering-induced distortion for the rendered view.

3.1.1. Packet-loss distortion model

For the RD-optimized transmission over error-prone network, the expected packet-loss-induced distortion for each encoding must be estimated prior to transmission. Assume that the retransmission is not supported due to its transmission delay, and the decoder can process the lost packet with an encoder-known error concealment operation. Thus, the distortion after error being concealed can be estimated at the server side. Also, in the current encoding, we packet one frame a packet and the decoder only conceals the whole frame loss.

In [24] and [25], the authors proposed the different distortion estimation models for IPPP coding structure. In our work, the first order distortion estimation (FODE) method [24] is used because it

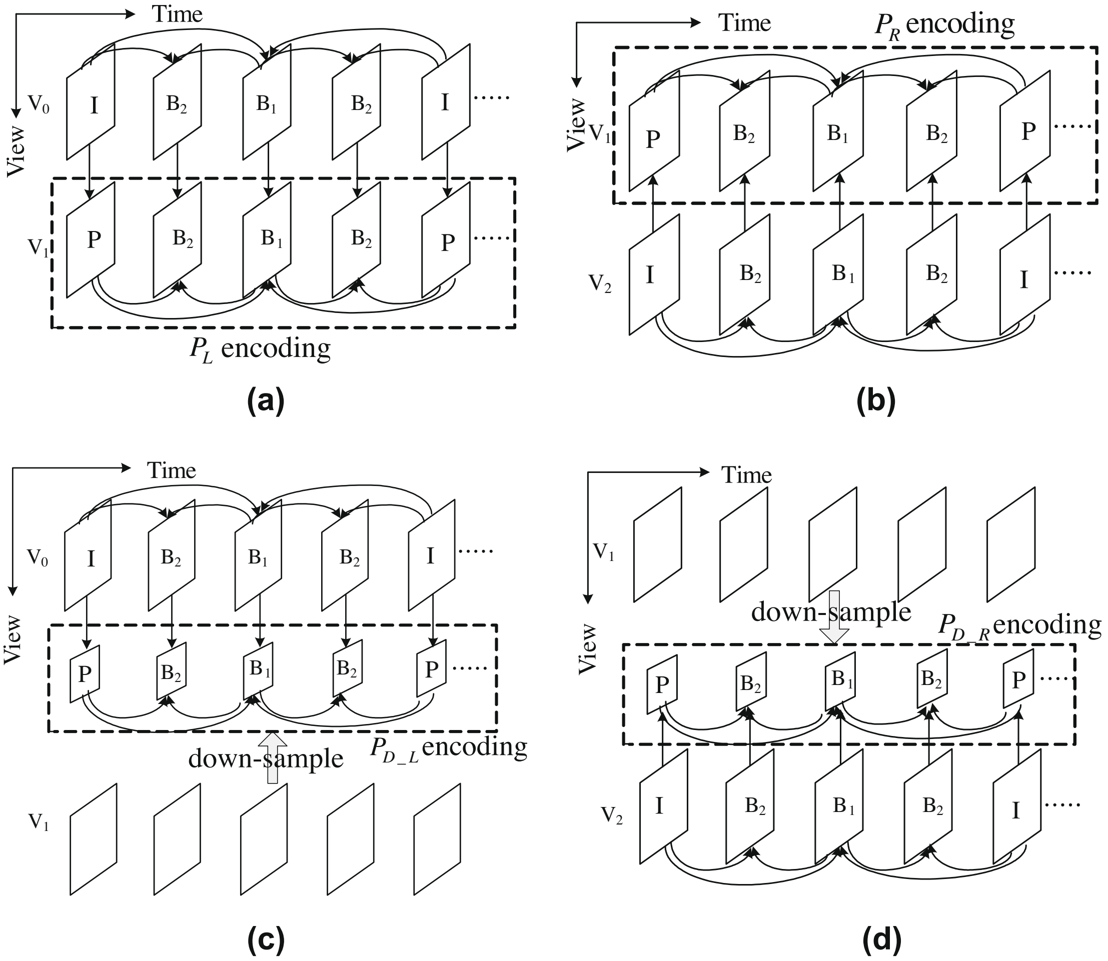


Fig. 3. Definitions of P_L, P_R, P_{D_L} and P_{D_R} encodings.

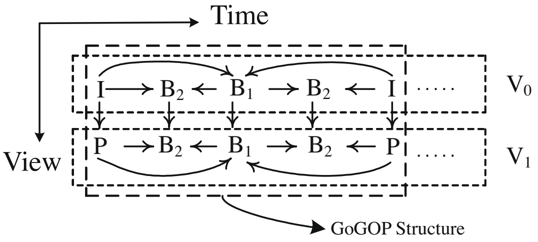


Fig. 4. Coding structure for one GoGOP.

is more suitable for the hierarchical B pictures coding structure with a short GOP size. Let $E\{D_{GOP}\}$ denote the expected distortion of one GOP, which is approximated in FODE by its first order Taylor expansion. With the packet loss rate p, $E\{D_{GOP}\}$ is estimated as

$$E\{D_{GOP}\} \approx D_{no\_loss} + p \cdot \sum_{i=0}^{L-1} \gamma_i \tag{4}$$

with

$$\gamma_i = D_{i\_loss} - D_{no\_loss}. \tag{5}$$

In (4), L is the GOP length, and D_{no_loss} means the distortion of the whole GOP without packet loss. In (5), γ_i is the first order Taylor expansion coefficient of frame i and D_{i_loss} denotes the GOP distortion due to only losing frame i. For the encodings of I, P_{D_L} and P_{D_R} in Fig. 3, some RD information should be kept in the server. The saved RD information includes the mean squared error (MSE) and the corresponding rate data of single frame loss event for L frames in a GOP.

For multiple encodings structure, the amount of the saved RD data is proportional to the number of encodings. Because of the same packet loss pattern, the distortion fading behaviors of different encodings are similar for the same sequence. We integrate this property into the distortion model to reduce the amount of the saved MSE data. For the same view sequence, an error propagation scaling method is derived to estimate the corresponding distortions of other encodings using the distortions of I encoding for all loss cases.

The distortion fading behavior is closely related with the coding structure. Here, the distortion fading denotes that the error propagation will gradually become weaken after the increased intra-block coding and the repeated encoder spatial filtering (i.e., sub-pixel interpolation). With the hierarchical prediction dependency between frames, distortion estimation should be classified into two cases, namely the single frame loss distortion estimations at the higher and the lowest hierarchical levels. For simplicity, we only discuss the distortion estimation of encoding P_L, and the estimation for P_R is the same.

We first discuss the single frame loss case at the higher level, for example, the case of losing B_1 frame for view1 in Fig. 5. Let D_{B1_P} and D_{B1_I} denote the distortions of encodings P_L and I for only losing B_1 frame, respectively. $D_{no\_loss\_I}$ and $D_{no\_loss\_P}$ represent the distortions of encodings I and P_L without any frame loss,

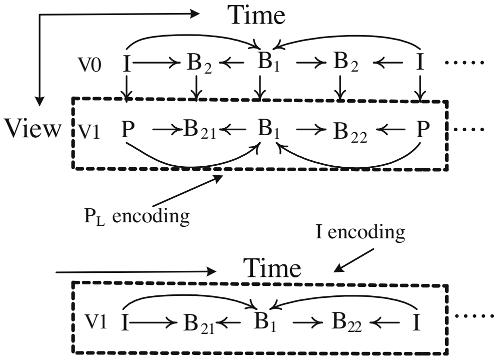


Fig. 5. Coding structures of different encodings.

respectively. Because of the similar error propagation, D_{B1_P} is not necessary to be saved and can be estimated from D_{B1_I} as

$$D_{B1\_P} \approx \begin{cases} (D_{B1\_I} - D_{no\_loss\_I}) + D_{no\_loss\_P}, & \text{if the current frame is } B_1, \\ (1-\alpha) \bullet (D_{B1\_I} - D_{no\_loss\_I} + D_{no\_loss\_P}) & \text{otherwise} \end{cases} \tag{6}$$

where α is a scaling factor limited in $(-1, 1]$. It is determined by the combination action of inter-view prediction, inter-frame prediction, and the encoder spatial filtering. The specific spatial filtering includes the sub-pixel interpolation filtering in inter-frame motion compensation and loop-filtering for reducing artifacts.

The spatial filtering can reduce the subsequent error propagation effect caused by the lost frame. The effects of spatial filtering for P_L encoding and I encoding are expressed as ϕ_P and ϕ_I, which account for the distortion attenuation factor for P_L and I encodings, respectively. For example in Fig. 5, the effect of lost B_1 frame in I encoding will spread to the frames B_{21} and B_{22}. However, the spatial filtering of B_{21} and B_{22} will mitigate this kind of error propagation.

Since P_L encoding introduces the inter-view prediction, the corresponding inter-frame prediction proportion which comes from the lost frame will be different from that of I encoding. Hence, the error propagation effects between the two encodings can be scaled via the different inter-frame prediction proportions which come from the lost frames. α is computed by

$$\alpha \approx 1 - \frac{\theta_P}{\theta_I} \cdot \frac{\varphi_P}{\varphi_I}, \tag{7}$$

where θ_I and θ_P denote inter-frame prediction proportion from the lost frame for I and P_L encodings, respectively. The percentage of inter-frame prediction in each frame is statistically recorded during encoding. Fig. 6 shows the specific θ_I and θ_P for B_1 frame in the first GOP for *Race1* sequence. For simplicity, $\frac{\varphi_P}{\varphi_I}$ in (7) is often set to a fixed value depending on the specific sequence.

For the single frame loss case at the lowest level, such as losing B_{22} for view1 in Fig. 5, the distortion estimation is the same with the lost frame case in (6). The distortions of the hybrid down-sampled encodings, by comparison, cannot be accurately estimated, because their distortions have different drifting properties from those of corresponding full-resolution encodings. As a result, we have to save the distortions of encodings P_{D_L} and P_{D_R}.

3.1.2. Rendering distortion model

In this section, we discuss how to estimate the end-to-end distortion for a rendered view. The RD analysis for a rendered light field is first given by [23], and the authors regard rendering a particular view as a linear combination model. In [26], a simple scaling method is presented to model the relationship of rendering error and angle of cameras. It is based on the rendering method with weighting rays which can achieve better efficiency for views with different angles. Generally, the scaling method is a very coarse approximation for rendering the practical scene [26]. Since the disparity-compensated view interpolation is an efficient view rendering method for multiview sequences with parallel camera setup, we model rendering as a disparity-compensated interpolation process. To estimate the rendering view distortion, we take into account not only the transferred coding and transmission distortions of adjacent source views, but also the post-filtering induced rendering distortion, such as the blending effect and pixel mapping uncertainty impact.

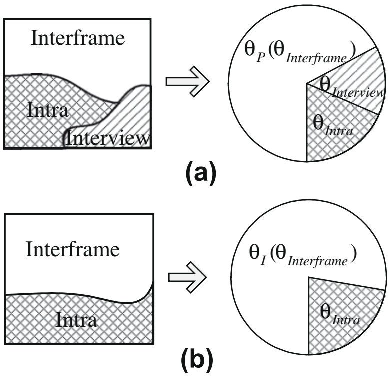


Fig. 6. θ_I and θ_P for *Race1* sequence: (a) θ_P for the frame predicted from the lost frame, (b) θ_I for the frame predicted from the lost frame. ("Interframe", "Interview" and "Intra" denote the proportions of inter-frame prediction, inter-view prediction and intra prediction for coding the frame, respectively.)

Block-based disparity compensation rendering for a block v_b of the rendered view v_r is expressed as $v_b = \sum_{i=1}^{N_r} w_i c_{b,i}$, where w_i is the rendering weight for the i^{th} source view, $c_{b,i}$ is the source block in the i^{th} view and N_r is the number of sources reference views for rendering. In order to accelerate rendering, we perform the fixed-size 8 × 8 block disparity compensation rendering.

In the block-based disparity compensation rendering, the artifacts are often occurred in the boundaries of blocks due to the block blending process. In order to reduce the artifacts, the rendered image needs to be post-filtered. Hence, the frame-level rendering can be approximately modeled as

$$v_r = \sum_{i=1}^{N_r} (w_i c_i + c_f), \tag{8}$$

where c_f denotes the post-filtering effect and c_i denotes the i^{th} source view.

Accordingly, the distortion of a rendered view at the n^{th} time instant is denoted by $D_{r,n}$, which can be modeled as

$$\begin{aligned} D_{r,n} &= E\left\{(v_{r,n} - \tilde{v}_{r,n})^2\right\} \\ &= E\left\{\left(\left(\sum_{i=1}^{N_r} w_{i,n} c_{i,n} + c_f\right) - \left(\sum_{i=1}^{N_r} \tilde{w}_{i,n} \tilde{c}_{i,n} + \tilde{c}_f\right)\right)^2\right\} \\ &= E\left\{\left(\sum_{i=1}^{N_r} w_{i,n} c_{i,n} - \left(\sum_{i=1}^{N_r} \tilde{w}_{i,n} \tilde{c}_{i,n} + \hat{c}_r\right)\right)^2\right\}, \end{aligned} \tag{9}$$

where $v_{r,n}$ is the rendered view using the original videos and $\tilde{v}_{r,n}$ is rendered view using the reconstructed videos. $w_{i,n}$ and $\tilde{w}_{i,n}$ signify the weight of view i using the original and reconstructed videos, respectively. Since $w_{i,n}$ and $\tilde{w}_{i,n}$ are related to the distance between the rendering view and adjacent source views, $w_{i,n}$ is approximately equal to $\tilde{w}_{i,n}$. Likewise in (9), c_f and $\tilde{c}_f$ denote the post-filtering on the view rendered using the original and reconstructed images, respectively. $\hat{c}_r$ denotes the error signal of the rendering view due to the post-filtering. In (9), the effect $\hat{c}_r$ is decomposed into each source-view's $\hat{c}_{i,n}$, which can be regarded as the error signal reflected on source image $c_{i,n}$ due to the post-filtering on the rendered image. Thus, (9) can be evolved into

$$D_{r,n} = E\left\{\left(\sum_{i=1}^{N_r} w_{i,n}c_{i,n} - \sum_{i=1}^{N_r} \tilde{w}_{i,n}c_{i,n} + \sum_{i=1}^{N_r} \tilde{w}_{i,n}c_{i,n} - \sum_{i=1}^{N_r} \tilde{w}_{i,n}\tilde{c}_{i,n} - \sum_{i=1}^{N_r} \tilde{w}_{i,n}\hat{c}_{i,n}\right)^2\right\} = E\left\{\left(\sum_{i=1}^{N_r} \tilde{w}_{i,n}(c_{i,n} - \tilde{c}_{i,n} - \hat{c}_{i,n})\right)^2\right\}. \quad (10)$$

Because $\hat{c}_{i,n}$ is uncorrelated with $(c_{i,n} - \tilde{c}_{i,n})$, the intermediate term can be thrown off. Then (10) is changed into

$$D_{r,n} \approx \sum_{i=1}^{N_r} \tilde{w}_{i,n}^2 \left(E\left\{(c_{i,n} - \tilde{c}_{i,n})^2\right\} + E\left\{\hat{c}_{i,n}^2\right\}\right) = \sum_{i=1}^{N} \tilde{w}_{i,n}^2 D_{i,n} + \Delta D_{r,n} = \bar{D}_{r,n} + \Delta D_{r,n}, \quad (11)$$

where $D_{i,n}$ denotes the coding distortion of the i^{th} source view, and $\bar{D}_{r,n}$ is the rendering distortion before filtering. $\Delta D_{r,n}$ denotes the post-filtering introduced distortion for the rendered image.

In (8), the rendering using the original source images also needs post filtering. Let $\overset{\smile}{f}{}^i_{r,n}$ represent the pixel value before filtering and $\overset{\frown}{f}{}^i_{r,n}$ denote the pixel value after filtering. The post-filtering is an average processing and it is assumed that the filtered pixel is not reused in the filtering. The filtering operation is mathematically modeled as

$$\overset{\frown}{f}{}^i_{r,n} = \sum_l \psi_l \overset{\smile}{f}{}^{l(i)}_{r,n} \quad \text{with} \quad \sum_l \psi_l = 1. \quad (12)$$

The index $l(i)$ represents the l^{th} neighboring pixel used in the filtering for pixel i. The set l includes the pixels used for filtering. The filter coefficients ψ_l are location and content dependent.

In the receiver rendering, the same filtering is also processed on the decoded pixels. Assume that the filter coefficient is identical to that of the filtering in the original source rendering. Let $\bar{f}^i_{r,n}$ be the reconstructed value for the i^{th} pixel before filtering, and $\tilde{f}^i_{r,n}$ after filtering, then

$$\tilde{f}^i_{r,n} = \sum_l \psi_l \bar{f}^{l(i)}_{r,n}. \quad (13)$$

In terms of (12) and (13), the filtering introduced distortion for a rendered frame is

$$\Delta D_{r,n} = E\left\{\left(\overset{\frown}{f}{}^i_{r,n} - \tilde{f}^i_{r,n}\right)^2\right\} = \sum_l \psi_l^2 E\left\{\left(\overset{\smile}{f}{}^{l(i)}_{r,n} - \bar{f}^{l(i)}_{r,n}\right)^2\right\} + \sum_{l,k,l\neq k} \psi_l\psi_k E\left\{\left(\overset{\smile}{f}{}^{l(i)}_{r,n} - \bar{f}^{l(i)}_{r,n}\right)\left(\overset{\smile}{f}{}^{k(i)}_{r,n} - \bar{f}^{k(i)}_{r,n}\right)\right\} = \left(\sum_l \psi_l^2 + \rho \sum_{l,k,l\neq k} \psi_l\psi_k\right)\bar{D}_{r,n} = \theta_n \bar{D}_{r,n}, \quad (14)$$

where ρ is the average correlation coefficient of the error between two neighboring pixels. Depending on the sign and magnitude of ρ, $\Delta D_{r,n}$ can either decrease or increase the rendering distortion. ρ cannot be obtained at server side and it is only approximately computed by source images. ρ is pre-computed in a 8 × 8 window for each source view image and then the average ρ in all source views is used. Combine (11) with (14) into a complete expression for the rendering distortion estimation as

$$D_{r,n} = (\theta_n + 1)\bar{D}_{r,n}, \quad (15)$$

where θ_n is a frame-dependent distortion scaling factor. According to the known filtering coefficients and ρ, θ_n can be computed and then the rendering distortion is appropriately estimated from (15). When packet loss occurs, $D_{i,n}$ in (11) is the estimated distortion which includes the packet-loss-induced distortion.

Since the distortion model in (9) is based on the assumption that the rendered image using the original source images is the original signal for a rendered image, $D_{r,n}$ is not the actually measured distortion of the rendered image and it is just a coarse approximation for the actually measured distortion. Nevertheless, this model can characterize how much the quantized distortion of one source view had been contributed to the rendered image so that $D_{r,n}$ can be used to measure the visual quality of a rendered image.

3.1.3. Distortion model accuracy

For packet-loss distortion estimation, FODE can provide accurate estimation at low to medium packet loss rate (PLR) [24]. Here, we only illustrate the distortion scaling accuracy by an example for the packet-loss distortion estimation. Fig. 7 shows the estimated result of encoding P_L using encoding I for *Race1* view0 sequence (640 × 480, 30 fps) when B_1 frame is lost in Fig. 5. It can be observed that the absolute error between the computed and estimated distortion is very small. The computed distortion is the actually computed MSE for P_L encoding. For the frames from 220 to 280, because the interview prediction proportions for the two encodings of I and P_L are very small, they have so little difference before and after error propagation that the estimation gives larger estimation error than other frames.

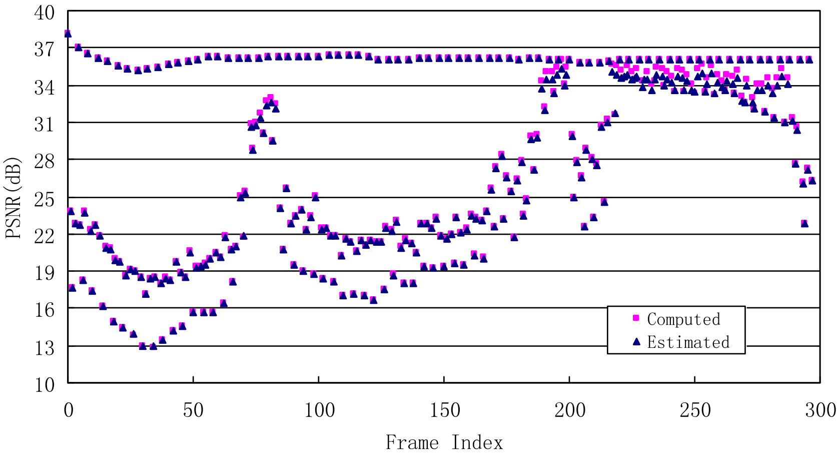


Fig. 7. P_L distortion estimation accuracy using I encoding (scatter diagram).

From the different encodings, we can synthesis different image versions for the same viewpoint. Fig. 8 shows that different rendered images using different encoding combinations. The rendering is operated at the middle position of the first frame of view6 and view7. Fig. 8(a) gives the rendered full-resolution picture (left) and the magnified object in the rendered scene (right). In Fig. 8(b)–(f), the images are magnified objects. As shown in Fig. 8, the images of (b)–(d) have the similar visual quality because the source images have little distortion difference between their encoding combinations. Since the images of (e) and (f) are rendered using down sampled source images, they exhibit the blurry visual effects compared with those of (b)–(d). In Fig. 8, the distortion is evaluated by MSE. The computed distortion using model (10) are described as D_C and the estimated distortion using model in (15) are expressed as D_E. Since our distortion estimation only involves identification of the optimum encoding combination, a certain degree of inaccuracy in the estimated distortion can be tolerated. It is noted from Fig. 8 that, D_E can characterize the rendering quality for a rendered image, and correspondingly it can be used as a perceptual distortion measurement for RD evaluation of the rendering process though it is a rough approximation to D_C.

3.2. RD-optimized scheduling algorithm

The compressed video data for one encoding combination are packed into independent GoGOPs. Assume that the system uses the M combinations for one GoGOP which includes N frames and each frame is packed into one packet. M is searched by sweeping all permutations and combinations which match the requested viewpoint. For the packet j in the i^{th} combination, the rate is $R_{i,j}$ and the expected end-to-end distortion is $E\{D_{i,j}\}$. When considering rendering-induced distortion, the expected distortion for packet j is computed as the specific form of (15). That is

$$E\{D_{i,j}\} = \tilde{w}_{i,j}^2(\theta_{i,j}+1)D_{i,j} \tag{16}$$

where $\tilde{w}_{i,j}$ and $\theta_{i,j}$ denote the rendering weight and the distortion scaling factor of the packet j in the i^{th} combination, respectively. $D_{i,j}$ is the distortion for the packet j in the i^{th} combination. Using R–D function

$$J_{i,j} = E\{D_{i,j}\} + \lambda_{i,j}R_{i,j}. \tag{17}$$

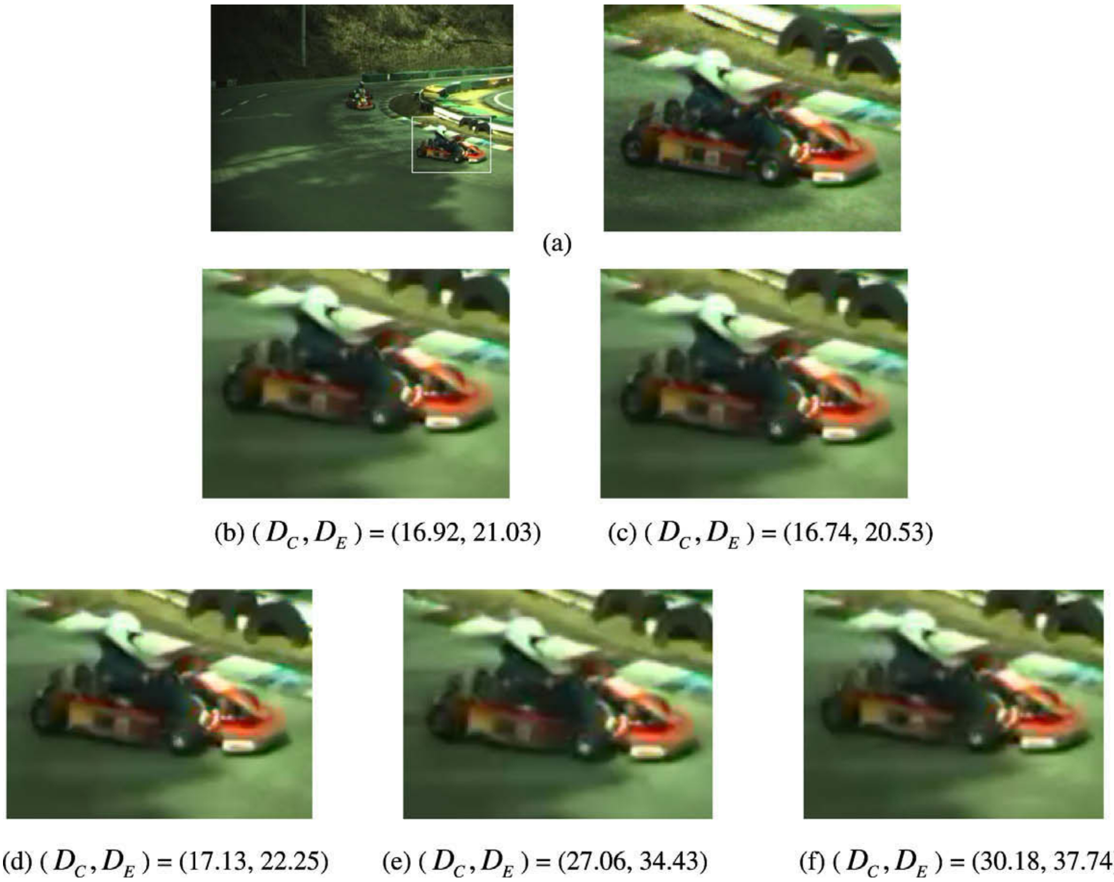
RD cost for packet j is achieved. Here, $\lambda_{i,j}$ is identical to that in H.264. The minimal RD cost for the GoGOP of combination i is calculated by

$$J_i = J_{i,0} + J_{i,1} + \cdots + J_{i,N-1}. \tag{18}$$

At last, the optimal combination is selected by $\hat{i} = \arg\min_{0<=i<M} J_i$.

Let V_{rq} denote the request view, V_{cp} denote the captured view, V_r denote the rendered view, and i_I represent one GOP of I encoding. PLR denotes the packet loss rate. The detailed RD-based scheduling algorithm is described as follows.

```
1. Initialize $V_{rq}$ and PLR.
2. if $V_{rq} = V_{cp}$ then $\hat{i}_{opt} = i_I$ and goto 1; else $V_{rq} = V_r$ and compute $M$
3. For one GoGOP
   for $i = 1$ to $M$ do
      for $j = 1$ to $N$ do
         compute $J_{i,j}$ according to (4), (16), and (17);
      compute $J_i$ according to (18);
   $\hat{i}_{opt} = \arg\min_{0<=i<M} J_i$.
4. If (the stream ends) then stop; else goto 1.
```



(a)

(b) $(D_C, D_E) = (16.92, 21.03)$

(c) $(D_C, D_E) = (16.74, 20.53)$

(d) $(D_C, D_E) = (17.13, 22.25)$

(e) $(D_C, D_E) = (27.06, 34.43)$

(f) $(D_C, D_E) = (30.18, 37.74)$

Fig. 8. The whole image or part image rendered with (a) original source images, (b) both I encodings for view6 and view7, (c) I for view6 and P_L for view7, (d) P_R for view6 and I for view7, (e) P_{D_R} for view6 and I for view7, (f) I for view6 and P_{D_L} for view7.

In the above algorithm, the complexity is about $O(M{\cdot}N)$. Since M and N are not very large numbers, the complexity is very low. Based on the RD-optimized scheduling algorithm, the optimal encodings are selected to adapt to the user's request as well as keep the optimal streaming performance.

4. Experimental results

4.1. End-to-end streaming performance

This section evaluates the end-to-end performance of the proposed streaming method. The sequence used in the experiment is *Race1* (640 × 480, 30 fps) provided by KDDI lab [27]. Eight views of each view with 300 frames are used. The error concealment method is the temporal frame copy technique which is added in the MVC software JMVM3.0.2 [28]. Packet loss patterns with a set of PLRs of 5%, 10% and 20% [29] are utilized to simulate the error-prone conditions. In the experiment, the bandwidth expansion effects of error-resilience channel coding, such as the additional rate cost of forward error correction (FEC), are ignored. Since the proposed streaming method aims at multiview video service with low-delay viewpoint switching at intra frame, the sequences are encoded with GOP size of four frames. In PSNR computing, since the original signal does not exist for a rendered image, it is replaced by the image rendered by the original source images.

To evaluate the end-to-end performance, two viewpoint motion trajectories are studied, as shown in Fig. 9. The first trajectory is a moderate viewpoint motion trajectory that the browsing-viewpoint stays at each view or the middle position of two views about 20 frames. When the browsing-viewpoint is at the position between two views, the virtual view is interpolated by disparity-compensated rendering. The second trajectory is a fast viewpoint motion trajectory with the uneven viewpoint switching.

In the experiments, two anchors are used for comparison. The first is selective streaming method using SMVC [19]. We adopt the SMVC3 structure which codes the base layer with three down-sampled views using the MVC and just one enhanced layer coded by H.264/AVC. The second anchor is view-dependent selective streaming of simulcast coding of each view. To compare the performances of the three methods, namely multiple encodings, simulcast coding and SMVC, hierarchical B pictures based open GOP coding structures are used.

To study the low-delay viewpoint switching, two transmission schemes are taken into account. The first is transmitting one view data when the browsing-viewpoint is at the original capturing position and transmitting two views while the browsing-viewpoint is at the middle position of two views. The second is always transmitting three views including one browsing view with two down-sampled side-views. Transmitting three views can reduce switching delay when the requested view is not arrived in time.

We first evaluate the first trajectory with moderate viewpoint motion. Fig. 10(a) gives the end-to-end RD performances of the three streaming methods without packet loss and with PLR $p = 10\%$ for the first transmission scheme, and Fig. 10(b) gives the end-to-end RD performances of the three methods for different cases for the second transmission scheme. In these figures, SC and SMVC denote the viewpoint-dependent selective transmission with simulcast coding and SMVC without packet loss, respectively. ME means the proposed view-dependent RD optimized streaming with multiple encodings without packet loss. SC_P=10%, SMVC_P=10% and ME_P=10% denote the three methods with $p = 10\%$, respectively.

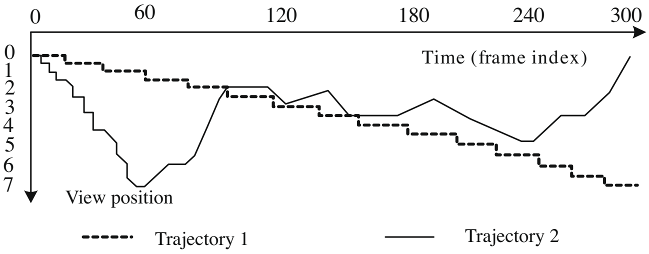


Fig. 9. Moving trajectories of the browsing-viewpoint.

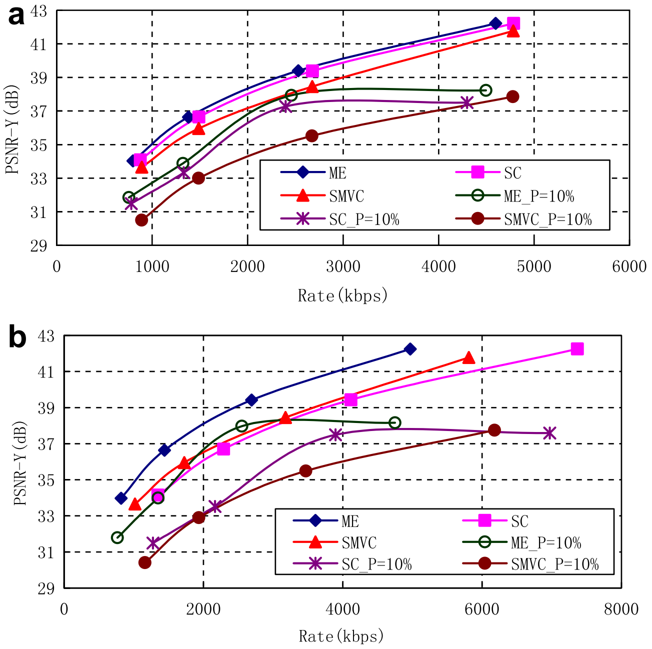


Fig. 10. End-to-end RD performance comparison for a moderate viewpoint motion trajectory. (a) First transmission scheme. (b) Second transmission scheme.

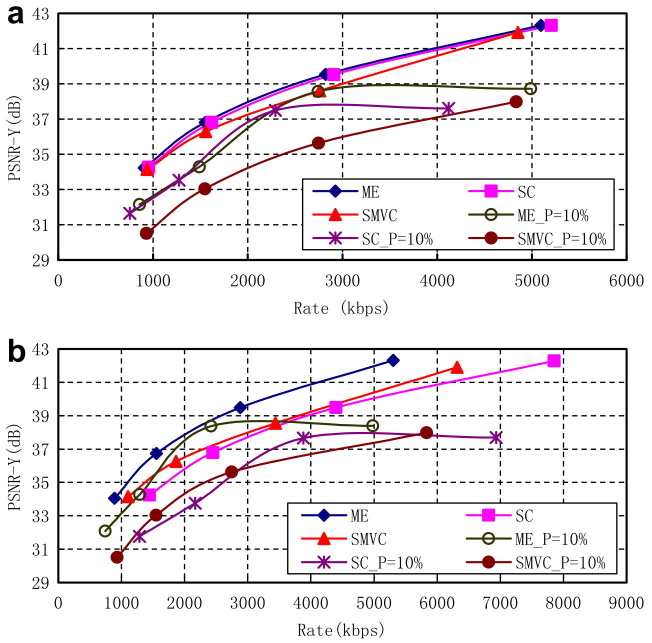


Fig. 11. End-to-end RD performance comparison for a fast viewpoint motion trajectory. (a) First transmission scheme. (b) Second transmission scheme.

As shown in Fig. 10, the RD-optimized interactive streaming with multiple encodings achieves the best performance among the three methods for error-free transmission. In Fig. 10(a), for packet-loss transmission, RD-optimized streaming of multiple encodings outperforms the simulcast selective streaming up to 0.8 dB and is also about 3 dB higher than SMVC selective streaming. In Fig. 10(b), the RD performance gap between the proposed method and other two methods becomes larger than that in Fig. 10(a) for error-prone streaming, and it is up to 4 dB for some cases. The comparisons illustrate that the proposed RD-optimized streaming method is more robust.

For the fast viewpoint motion trajectory, Fig. 11 gives the performance comparison of several selective streaming methods. It can be observed that, whether the channel has packet loss or no packet loss, the RD-optimized streaming with multiple encodings has always the better performance than other two methods. In Fig. 11(a), at low bit rate, the multiple encodings streaming has similar performance with the simulcast coding streaming. However, at high bit rates, the multiple encodings streaming achieves about 1 dB gain than simulcast coding streaming. Fig. 11(b) shows that the multiple encodings method is also superior to other two methods, especially for packet loss cases.

From Fig. 10 and Fig. 11, it is worthwhile to be noted that the performance of the proposed method changes very little with the moving velocity of browsing-viewpoint. In other words, RD-optimized multiple encodings streaming can achieve a higher performance in saving channel resource and flexible view switching. In the application, the continuous browsing in the viewpoint dimension is needed. When the viewpoint is selected on the position between the two captured views, multiple encodings can save a lot of rates because of the inter-prediction among the transmitted views. The performance of multiple encodings streaming is somewhat related to the browsing trajectory. The longer the viewpoint rests on one position between the two captured views, the more gains multiple encodings streaming can obtain.

Fig. 12 shows the PSNR performances at different PLRs for different streaming methods with a moderate viewpoint motion trajectory. It indicates that the ME streaming has the best RD performance among the three streaming methods. The SMVC method exhibits the inferior performance at the cases of different PLRs. In the packet loss pattern provided by [29], there are very few burst loss cases. To investigate the RD performance of the proposed multiple encodings streaming with more burst loss cases, we revise the packet loss pattern to increase the burst loss cases up to 30% in all loss cases of each packet loss pattern. Fig. 13 shows the streaming performance comparison among the different methods with a moderate viewpoint motion trajectory under more burst loss cases. With the more burst loss cases, the RD performances of all the methods are much worse. However, ME streaming can achieve the relatively higher performance than the other two selective streaming methods

4.2. Server complexity

Bandwidth is an expensive resource for video streaming application. The video streaming with multiple encodings saves the channel resource at the price of more server storage. If K encodings are used, the storage will increase $K-1$ server-storage compared with traditional one encoding streaming. Since K is not a very large number, the streaming method with multiple encodings is acceptable.

Besides the storage, multiple encodings bring more off-line processing complexity. The computation of γ_i in FODE involves $2L{\cdot}K$ times distortion calculations for K encodings (about $2L{\cdot}K$ times decoding), and L is the GOP size. In our experiments, K is set to 5 and L is set to 4 for low delay application. The distortion calculation for multiple encodings is greatly reduced due to the adoption of distortion scaling.

For RD-optimized streaming, the server incurs some computation in real-time scheduling, which makes the multiple encodings

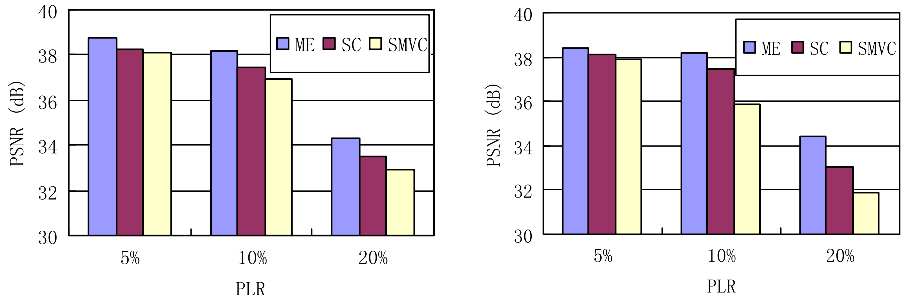


Fig. 12. The comparisons of PSNR vs. PLR for several streaming schemes. (Bit-rate is at 3800kbps: (a) First transmission scheme, (b) Second transmission scheme.)

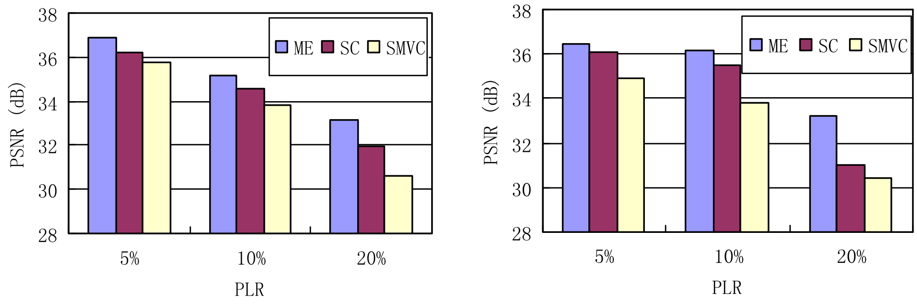


Fig. 13. The comparisons of PSNR vs. PLR for several streaming schemes with more burst loss cases. (Bit-rate is at 3800kbps: (a) First transmission scheme, (b) Second transmission scheme.)

streaming a bit more complex than simulcast coding method. However, compared with SMVC method, which needs rate allocation computation to stream the multiview video with multiple layers, multiple encodings method is of approximately similar complexity.

5. Conclusion

Interactive streaming using emerging H.264/MVC standard is a challenging problem for multiview services. In this paper, we present an interactive, view-dependent streaming method with multiple encodings structure to spur H.264/MVC standard application. This method is a practical solution to save transmission bandwidth for interactive multiview video services. The proposed method can provide the superior interactivity using multiple encodings. Via taking into account the different kinds of distortion, the proposed RD-optimized scheduling strategy can guarantee the optimal performance by taking full advantage of the RD characteristics of multiple encodings. The proposed method can provide better error-resilient end-to-end RD performance than streaming methods with SMVC and simulcast coding.

Acknowledgments

This work is supported in part by National Natural Science Foundation Research Program of China under Grant Nos. 60736043 and 60833006, National Basic Research Program of China (973 Program, 2009CB320905) and National Hi-Tech R&D Program (863 Program) under contract No. 2007AA01Z315. The authors would like to thank KDDI for providing the video sequence of *Race1*.

References

[1] H. Kimata, M. Kitahara, K. Kamikura, Y. Yashima, T. Fujii, M. Tanimoto, Low delay multi-view video coding for free-viewpoint video communication, IEICE Jpn. J89-J (1) (2006) 40–55.
[2] M. Tanimoto, Overview of free viewpoint television, Signal Process. Image Commun. 21 (6) (2006) 454–461.
[3] J. Luo, H. Cai, J. Li, A real-time interactive multiview video system, in: Proc. of the 13th ACM International Conference on Multimedia, 2005, pp. 161–170.
[4] A. Smolic, P. Kauff, Interactive 3D video representation and coding technologies, Proc. IEEE (2005) 98–110.
[5] A. Vetro, W. Matusik, H. Pfister, J. Xin, Coding approaches for end-to-end 3D TV systems, in: Proc. of Picture Coding Symposium (PCS), Dec. 2004.
[6] J.H. Kim, P. Lai, J. Lopez, A. Ortega, Y. Su, P. Yin, C. Gomila, New coding tools for illumination and focus mismatch compensation in multiview video coding, IEEE Trans. Circuits Syst. Video Technol. 17 (11) (2007) 1519–1535.
[7] P. Merkle, A. Smolic, K. Müller, T. Wiegand, Efficient prediction structures for multiview video coding, IEEE Trans. Circuits Syst. Video Technol. 17 (11) (2007) 1461–1473.
[8] S.-U. Yoon, Y.-S. Ho, Multiple color and depth video coding using a hierarchical representation, IEEE Trans. Circuits Syst. Video Technol. 17 (11) (2007) 1450–1460.
[9] W. Yang, Y. Lu, F. Wu, J. Cai, K.-N. Ngan, S. Li, 4-D wavelet-based multiview video coding, IEEE Trans. Circuits Syst. Video Technol. 16 (11) (2006) 1385–1396.
[10] Y. Liu, Q. Huang, X. Ji, D. Zhao, W. Gao, Multiview video coding with flexible view-temporal prediction structure for fast random access, in: Proc. of Pacific-Rim Conference on Multimedia (PCM), 2006, pp. 564–571.
[11] Y. Liu, Q. Huang, D. Zhao, W. Gao, Low-delay view random access for multi-view video coding, in: Proc. of IEEE International Symposium on Circuits and Systems, 2007, pp. 997–1000.
[12] A.M. Tekalp, E. Kurutepe, M.R. Civanlar, 3DTV over IP, IEEE Signal Process. Mag. 24 (6) (2007) 77–87.
[13] A. Mavlankar, D. Varodayan, B. Girod, Region-of-interest prediction for interactively streaming regions of high resolution video, in: Proc. of IEEE International Packet Video Workshop (PV), 2007, pp. 68–77.
[14] J. Li, H. Sun, On interactive browsing of large images, IEEE Trans. Multimedia 5 (4) (2003) 581–590.
[15] C. Zhang, J. Li, On the compression and streaming of concentric mosaic data for free wondering in a realistic environment over the internet, IEEE Trans. Multimedia 7 (6) (2005) 1170–1182.
[16] P. Ramanathan, M. Kalman, B. Girod, Rate-distortion optimized interactive light field streaming, IEEE Trans. Multimedia 9 (4) (2007) 813–825.
[17] J. Lou, H. Cai, J. Li, Interactive multiview video delivery based on IP multicast, Adv. Multimedia, vol. 2007, Article ID 97535.
[18] E. Kurutepe, M.R. Civanlar, A.M. Tekalp, Interactive transport of multiview videos for 3DTV applications, J. Zhejiang Univ.: Science A 7 (5) (2006) 830–836.
[19] E. Kurutepe, M.R. Civanlar, A.M. Tekalp, Client-driven selective streaming of multiview video for interactive 3DTV, IEEE Trans. Circuits Syst. Video Technol. 17 (11) (2007) 1558–1565.
[20] I. Radulovic, P. Frossard, O. Verscheure, Adaptive video streaming in lossy networks: versions or layers? in: Proc. of IEEE ICME, Taipei, June 2004.
[21] E. Kurutepe, M.R. Civanlar, A.M. Tekalp, Interactive multiview video delivery with view-point tracking and fast stream switching, in: Proc. of International Workshop on Multimedia Content Representation Classification and Security, 2006, pp. 586–593.
[22] Z. He, J. Cai, C. Chen, Joint source channel rate-distortion analysis for adaptive mode selection and rate control in wireless video coding, IEEE Trans. Circuits Syst. Video Technol. 12 (6) (2002) 511–523.
[23] P. Ramanathan, B. Girod, Rate-distortion analysis for light field coding and streaming, Signal Process. Image Commun. 21 (6) (2006) 462–475.
[24] R. Zhang, S.L. Regunathan, K. Rose, End-to-end distortion estimation for RD-based delivery of pre-compressed video, in: IEEE Conference Record of the Thirty-Fifth Asilomar Conference on Signals, Systems and Computers, 2001.
[25] J. Chakareski, J.G. Apostolopoulos, S. Wee, W. Tan, B. Girod, Rate-distortion hint tracks for adaptive video streaming, IEEE Trans. Circuits Syst. Video Technol. 15 (10) (2005) 1257–1269.
[26] C. Zhang, T. Chen, Active rearranged capturing of image-based rendering scenes – theory and practice, IEEE Trans. Multimedia 9 (3) (2007) 520–531.
[27] A. Ishikawa, KDDI test sequences for call for proposals on multiview video coding, in: MPEG2005/m12402, 73th MPEG Meeting.
[28] ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, Joint Multiview Video Model (JMVM) 3.0, 2007.
[29] M. Luttrell, S. Wenger, M. Gallant, New versions of packet loss environment and pseudomux tools, ITU-T VCEG Q15-I-09 (1999).