Spatial-Random-Access-Enabled Video Coding for Interactive Virtual Pan/Tilt/Zoom Functionality

Aditya Mavlankar, *Member, IEEE,* and Bernd Girod, *Fellow, IEEE*

***Abstract*—High-spatial-resolution videos offer the possibility of viewing an arbitrary region-of-interest (RoI) interactively. Zoom functionality enables watching high-resolution content even on displays of lower spatial resolution. If arbitrary regions corresponding to arbitrary zoom factors can be served to the user, the transmission and/or decoding of the entire high-spatial-resolution video can be avoided. Moreover, if the video content can be encoded such that arbitrary RoIs corresponding to different zoom factors can be simply extracted from the compressed bitstream, we can avoid dedicated video encoding for each user. We propose such a video coding scheme that is vital in allowing the system to scale to large numbers of remote users as well as to encode and store the content for subsequent repeated playback. Apart from generating a multi-resolution representation, our coding scheme uses P slices from H.264/AVC. We study the tradeoff in the choice of slice size. A larger slice size enables higher coding efficiency for representing the entire scene but increases the number of pixels that have to be transmitted. The optimal slice size achieves the best tradeoff and minimizes the expected transmission bitrate. Experimental results confirm the optimality of our predicted slice size for various test cases. Furthermore, we propose an improvement based on background extraction and long-term memory motion-compensated prediction. Experiments indicate up to 85% bitrate reduction while retaining efficient random access capability.**

***Index Terms*—Interactive video streaming, pan/tilt/zoom, region-of-interest.**

I. Introduction

HIGH-spatial-resolution digital video will be widely available at low cost in the near future. This development is driven by increasing spatial resolution offered by digital imaging sensors and increasing capacities of storage devices. Furthermore, there exist algorithms for stitching a comprehensive high-resolution view from multiple cameras [1], [2]. Certain current products stitch a large panoramic view in real time [3]. Also, image acquisition on spherical, cylindrical, or hyperbolic image planes via multiple cameras can record scenes with a wide field-of-view while the recorded data can be warped later to the desired viewing format [4]. An example of such an acquisition device is [5].

Despite the availability of high-resolution video, challenges in delivering this high-resolution content to the client are posed by the limited resolution of displays and/or limited data rate for communications. If the user were made to watch a spatially downsampled version of the entire video scene, then she might not be able to watch a local region-of-interest (RoI) with the recorded high resolution. To overcome this problem, we propose interactive virtual pan/tilt/zoom functionality while viewing the video. Some practical scenarios where this kind of interactivity is well-suited are: interactive playback of a high-resolution video from a locally stored file, interactive TV for watching content captured with very high detail (e.g., interactive viewing of sports events), providing virtual pan/tilt/zoom within a wide-angle and high-resolution scene from a surveillance camera, and streaming instructional videos captured with high spatial resolution (e.g., panel discussions, lecture videos). A video clip that showcases interactive viewing of soccer in a TV-like setting can be seen here [6].

In a streaming scenario, our proposed video coding scheme allows transmitting user-selected RoIs, thus eliminating the need to transmit the entire spatial extent of the scene in full resolution. The encoding can either take place live or offline beforehand. Additionally, our scheme allows limiting the load of encoding irrespective of the number of users. The entire recorded field-of-view can be encoded once, possibly with multiple resolution layers to support different zoom factors. Spatial resolution layers are coded using P slices[1] of H.264/AVC. This one-time encoding generates a repository of slices, and relevant slices can be served to several users depending on their individual RoIs. Thus, the coding scheme allows the system to scale to large numbers of users; it avoids a dedicated encoder for each user's individual RoI sequence. Another benefit is that requested RoIs can be extracted from the bitstream even inside or at the edge of the network, closer to the client-nodes. Ideally, the video delivery system should be able to react to the user's changing RoI with as little latency as possible. The proposed coding scheme enables access to a new region, with an arbitrary zoom factor, during any frame interval instead of having to wait for the end of a group of pictures (GoP) or having to transmit extra slices from previous frames.

Manuscript received May 21, 2009; revised October 22, 2009 and July 30, 2010; accepted October 18, 2010. Date of publication March 17, 2011; date of current version May 4, 2011. This paper was recommended by Associate Editor I. Ahmad.

A. Mavlankar was with Stanford University, Stanford, CA 94305 USA. He is now with Tely Labs, Inc., Menlo Park, CA 94025 USA (e-mail: aditya.mavlankar@ieee.org).

B. Girod is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305 USA (e-mail: bgirod@stanford.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/TCSVT.2011.2129170

[1]We employ the following terminology: "slice" refers to a rectangular portion of a video frame, whereas "tile" refers to the sequence of slices from the same resolution layer and at the same position in each video frame.

The spatial random access approach developed in this paper is also relevant for the design of systems that employ image-based-rendering (IBR) [7], [8] and manipulate the transmitted imagery further to yield a novel view, e.g., teleimmersive systems [9] and free viewpoint TV [10].

This paper is structured as follows. Section II reviews related work and discusses the challenges in providing random access. Section III presents the coding scheme and discusses how to optimize the slice size. The optimal slice size minimizes transmission bitrate by striking the best compromise between compression efficiency and superfluous pixel transmission. Section IV presents an improvement of the coding scheme based on background extraction and long-term memory motion-compensated prediction. Experiments indicate that the proposed improvement can reduce bitrate by up to 85% while retaining efficient random access capability.

II. Related Work

Taubman *et al.* [11] proposed a solution for interactive browsing of images using JPEG2000. The multi-resolution representation of an image using wavelets is leveraged to provide pan/tilt/zoom. JPEG2000 encodes blocks of wavelet transform coefficients independently. Consequently, every coded block has influence on the reconstruction of a limited number of pixels of the image. Moreover, the coding of each block results in an independent, embedded bitstream, which allows streaming any given block with a desired degree of fidelity. Taubman *et al.* also developed the JPEG2000 over Internet Protocol, for communication between client and server that supports remote interactive browsing of JPEG2000 coded images [12]. The server can keep track of the RoI trajectory of the client as well as the parts of the bitstream that have already been streamed to the client. Given a rate of transmission for the current time interval, the server solves an optimization problem to determine which parts of the bitstream should be sent in order to maximize the quality of the current RoI. This is similar to packet scheduling algorithms proposed in [13] for streaming of video. It should be noted, however, that an accurate model for the distortion reduction due to successful delivery of any particular packet is necessary.

Video coding for spatial random access presents a special challenge. To achieve good compression efficiency, video compression schemes typically employ motion-compensated interframe prediction for exploiting correlation among successive frames [14]–[16]. However, the coding dependencies among successive frames make it difficult to provide random access for spatial browsing within the scene. The decoding of a block of pixels requires that other reference frame blocks used by the predictor have previously been decoded. These reference frame blocks might lie outside the RoI and might not have been transmitted or decoded earlier.

Coding, transmission, and rendering of high-resolution panoramic videos using MPEG-4 is proposed in [17]. A limited part of the entire scene is transmitted to the client depending on the chosen viewpoint. In [17], only intraframe coding is used to allow random access. The scene is subdivided into slices which are coded independently. The authors also considered interframe coding to improve compression efficiency. However, they noted that this involves transmitting slices from the past if the current slice requires those for its decoding. A longer intraframe period entails significant transmission overhead for slices from the latter frames in the GoP, as this dependency chain grows. Besides the transmission overhead, the reference frame blocks also entail growing overhead of decoding.

Coding and streaming of images from an IBR representation also entails the random access issue associated with interframe coding. This applies both when the captured scene is static or evolving in time. Interactive streaming of static light fields has been studied by Ramanathan *et al.* in [18] and [19]. The above-mentioned growing dependency chain is avoided by using multiple representations coding based on two new picture types defined in the H.264/AVC standard, SP, and SI picture types [20]. Ramanathan *et al.* also extended rate-distortion optimized packet scheduling, based on the framework in [13], to multiple representations coding for light fields. However, in their setup, only entire pictures from the light field data-set are streamed and there is no provision of spatial random access within a picture. Compression and streaming of static light fields using distributed source coding has been investigated in [21] and [22]. If adequate rate is spent for signaling the non-key frame then identical reconstruction is guaranteed independent of the "reference blocks" used as side information at the receiver. Although this simplifies random access, the coding efficiency is lower than hybrid video coding and the problem of rate estimation while streaming is challenging. Bauermann *et al.* conducted a detailed analysis of the decoding complexity and the mean transmission bitrate for remote access to arbitrary parts of compressed image-based scene representations encoded using hybrid video coding [23], [24]. Their work, however, does not include a multi-resolution representation of the image data-set and is restricted to static imagery. Also, for saving transmission bitrate, apart from knowing which pixel blocks are currently required, the server also needs to know which pixel blocks have already been transmitted to the user. The server uses this information to stream a burst of reference pixel blocks. The variation of instantaneous bitrate and decoding load are undesirable.

Recently, Kurutepe *et al.* [25] proposed live interactive 3DTV based on dynamic light fields. They employed application-layer peer-to-peer (P2P) multicast and delivered a subset of views to a peer from a set of multiview videos of the scene. Similar to [18] and [19], entire views are either selected or dropped according to the peer's viewpoint. Random access to arbitrary views is provided by encoding the views independently. Multicasting lowers the bandwidth requirement at the server, however, the coded representation should consist of logical substreams for which multicast groups can be formed. Efficient random access is highly desirable since it simplifies the peer's task of deciding which multicast groups to subscribe. Similar to [18] and [19], entire frames from the data-set are streamed or not, and there is no provision of spatial random access within a picture.

Background extraction for motion-compensated prediction has been proposed in [26]. Sprite coding defined in MPEG-4

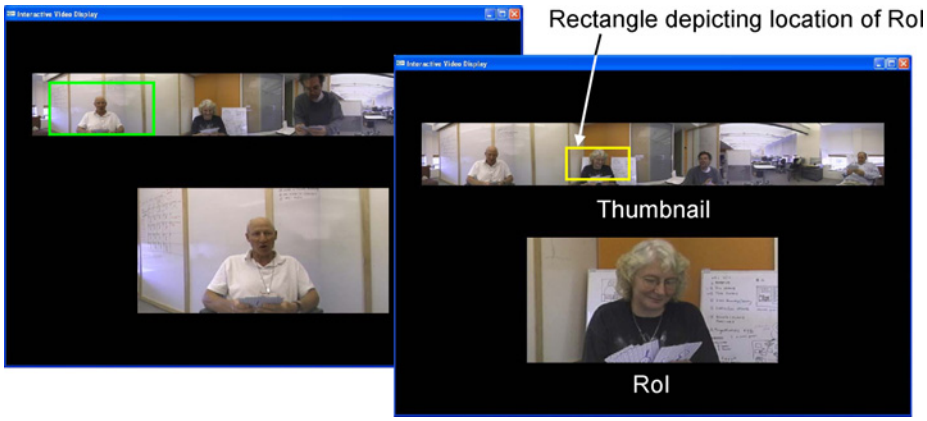

Fig. 1. Graphical user interface. The client's display shows the thumbnail and the RoI. The effect of changing the zoom factor can be seen by comparing the two screenshots. Each screenshot shows a frame of the panoramic *Cardgame* video sequence used in our experiments.

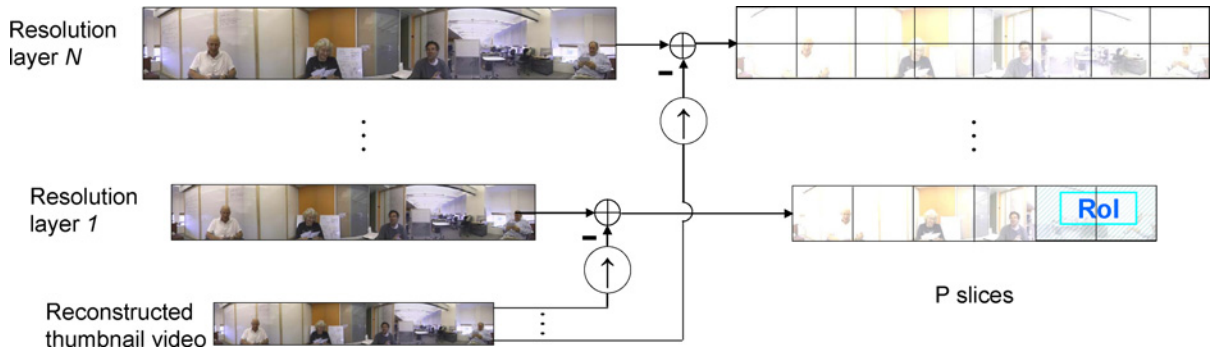

Fig. 2. Video coding scheme. The thumbnail video constitutes a base layer and is coded with H.264/AVC using I, P, and B pictures. The reconstructed base layer video frames are upsampled by a suitable factor and used as prediction signal for encoding video corresponding to the higher resolution layers. Higher resolution layers are coded using P slices.

Visual (MPEG-4 Part 2) allows coding the background either fully or partly for subsequent use as reference in predictive coding. The term "sprite" is more general and covers any transmitted video object that can be warped and/or cropped in certain ways for use by the motion predictor. However, unlike our proposed scheme, the compression schemes in the literature employing background extraction are not designed to provide virtual pan/tilt/zoom functionality.

III. Spatial-Random-Access-Enabled Video Coding

We have developed a graphical user interface which allows the user to select the RoI while watching the video. The RoI location and zoom factor are controlled by operating the mouse. The application supports continuous zoom to provide smooth control of the zoom factor. In addition to the RoI, we also display a thumbnail overview with an overlaid rectangle indicating the location of the RoI. Screenshots of the client's display are shown in Fig. 1.

A. Coding Scheme Based on Upward Prediction and Slices

Fig. 2 shows the video coding scheme. The thumbnail overview constitutes a base layer video and is coded with H.264/AVC using I, P, and B pictures. The reconstructed base layer video frames are upsampled by a suitable factor and used as prediction signal for encoding video corresponding to the higher resolution layers. Each frame belonging to a higher resolution layer is coded using a grid of rectangular P slices. Employing upward prediction from only the thumbnail enables efficient random access to local regions within any spatial resolution. For a given frame interval, the display of the client is rendered by transmitting the corresponding frame from the base layer and few P slices from exactly one higher resolution layer. We transmit slices from that resolution layer which corresponds closest to the user's current zoom factor. At the client's side, the corresponding RoI from this resolution layer is resampled to correspond to the user's zoom factor. We may store few spatial resolution layers at the server but can still render smooth zoom control. If a required enhancement layer P slice is unavailable at the client, for example, due to loss in the network, we perform error concealment by upsampling portions of the thumbnail video.

In our experiments, the spatial resolution layers stored at the server are dyadically spaced. Hence, the reconstructed thumbnail frame needs to be upsampled by powers of two horizontally and vertically to generate the corresponding prediction signals. For upsampling the luminance component, we employ the six-tap filter having the coefficients $(1, -5, 20, 20, -5, 1)/32$ as defined in H.264/AVC. For chroma, we employ a simple two-tap filter with equal coefficients. The upsampling procedure is repeated an appropriate number of times depending on the resolution layer. Although

we choose these parameters for our experiments, our design can incorporate arbitrarily spaced resolution layers and also arbitrary procedures for upsampling the reconstructed base layer. Also, at the client's side, for resampling the corresponding RoI from the chosen resolution layer, any technique can be accommodated. In our experiments, we use bilinear interpolation.

B. Comparison with Current Video Compression Standards

The coding scheme proposed above uses H.264/AVC building blocks but itself is not standard compliant. State-of-the-art video compression standards, H.264/AVC and SVC, provide tools like slices but no straightforward method for spatial random access since their main focus has been compression efficiency of full-frame video and resilience to losses. SVC supports both slices as well as spatial resolution layers. Alas, SVC allows only single-loop decoding whereas upward prediction from intercoded base-layer frames implies multiple-loop decoding, and hence is not supported by the standard. If the base layer frame is intercoded, then SVC allows predicting the motion-compensation residual at the higher-resolution layer from the residual at the base layer. However, interframe prediction dependencies across tiles belonging to a high-resolution layer hamper spatial random access. Note that for employing SVC, the motion vectors (MVs) can be chosen to avoid inter-tile dependencies. Also note that instead of SVC, AVC could be employed separately for the high-resolution layers with the MVs similarly restricted to eliminate inter-tile dependencies. This is very similar to treating the tiles as separate video sequences. An obvious drawback is the redundancy between the high-resolution tiles and the base layer. A second drawback is that after RoI change, a newly needed tile can only be decoded starting from an intracoded slice. However, note that B slices could also be employed for the high-resolution layers.

Prior work on view random access, discussed in Section II, employs multiple representations for coding an image. Similarly, we can use multiple representations for coding a high-resolution slice. This will allow us to use interframe coding among successive high-resolution layer frames and to transmit the appropriate representation for a slice depending on the slices that have been transmitted earlier. Some representations will exploit inter-tile correlation, thus lowering the transmission bitrate. However, more storage will be required for multiple representations. The benefit of the scheme in Fig. 2 is that knowing the current RoI is enough to decide which data need to be transmitted unlike the case of multiple representations where the decision is conditional on prior transmitted data.

In our proposed scheme, motion compensation among successive frames is performed at the base layer. We also employ displacement compensation with a small search range of about four pixels to find the best match relative to the upsampled base layer frame while coding the high-resolution P slices. The total encoding load is determined by the maximum resolution and the number of layers and can be estimated to be roughly 1.3 times the load of encoding just the highest resolution layer using standard motion-compensated hybrid video coding.

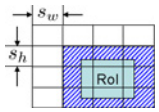

Fig. 3. Depending on the slice size and the location of the RoI within the given resolution layer, there is an overhead of pixels that are transmitted but not used for rendering the client's display. The shaded portion depicts the pixel overhead in this example.

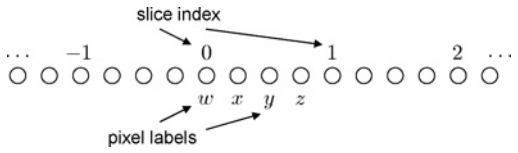

Fig. 4. Sequence of pixels is divided into 1-D "slices." In this example, the length of each slice is $s = 4$. The length of the 1-D "region-of-interest" is $R = 3$.

C. Minimization of Mean Transmission Bitrate

For the coding scheme shown in Fig. 2, the slice size for each resolution layer can be independently optimized given the prediction residual for that layer. The strategy proposed here can be independently used for all layers. Given a resolution layer, we assume that the slices form a regular rectangular grid, so that every slice is s_w pixels wide and s_h pixels tall. The slices on the boundaries can have smaller dimensions due to the layer dimensions not being integer multiples of the slice dimensions.

The number of bits transmitted to the client, or decoded for local playback, depends on the slice size as well as the user's RoI trajectory over the interactive viewing session. The quality of the decoded video depends on the quantization parameter (QP) used for encoding the slices. However, it should be noted that for the same QP, almost the same quality is obtained for different slice sizes, even though the number of bits is different. Hence, given the QP, our goal is to choose the slice size that minimizes the expected number of bits transmitted and/or decoded per rendered pixel. The smaller the slice size the worse is the coding efficiency. This is because of increased number of slice headers, lack of context continuation across slices for context adaptive coding, and inability to exploit inter-pixel correlation across slices. On the other hand, a smaller slice size entails lower pixel overhead. The pixel overhead consists of pixels that have to be transmitted and/or decoded because of the coarse slice division, but are not used to render the client's display. For example, the shaded pixels in Fig. 3 show the pixel overhead for the shown slice grid and location of the RoI.

In the following analysis, we assume that the RoI location can be changed with a granularity of one pixel both horizontally and vertically. Also, every location is equally likely to be selected. Depending on the application scenario, the slices might be put in different transport layer packets. The packetization overhead of layers below the application layer, for example RTP/UDP/IP, has not been taken into account but can be easily incorporated into the proposed optimization framework.

1) *Pixel Overhead:* To simplify the analysis, we first consider the 1-D case and then extend it to 2-D.

a) *Analysis in 1-D:* Imagine an infinitely long sequence of pixels. This sequence is divided into "slices" of length s. For example, in Fig. 4, $s = 4$. Also given is the length of the "region-of-interest," denoted by R. Assume $R = 3$ in this example. To calculate the pixel overhead, we are interested in the probability distribution of the number of 1-D "slices" that need to be transmitted. This can be obtained by testing for locations within one slice, since the pattern repeats every slice. For RoI locations w and x, we would need to transmit a single slice, whereas for locations y and z, we would need to transmit two slices. Let N be the random variable representing the number of slices to be transmitted. Given s and R, we can uniquely choose m, $R^* \in \mathbb{N}$ such that $m \geq 0$ and $1 \leq R^* \leq s$ and also the following relationship holds:

$$R = ms + R^*. \tag{1}$$

By inspection, we find the p.m.f. of random variable N

$$\begin{aligned} \Pr\{N = m+1\} &= \frac{s - (R^* - 1)}{s} \\ \Pr\{N = m+2\} &= \frac{R^* - 1}{s} \end{aligned}$$

and zero everywhere else. From the p.m.f. of N

$$\begin{aligned} E\{N\} &= (m+1)\frac{s - (R^* - 1)}{s} + (m+2)\frac{R^* - 1}{s} \\ &= (m+1) + \frac{R^* - 1}{s}. \end{aligned} \tag{2}$$

Let P be the random variable which denotes the number of pixels that need to be transmitted

$$\begin{aligned} E\{P\} &= s \times E\{N\} \\ &= (m+1)s + R^* - 1 \\ &= R + s - 1. \end{aligned} \tag{3}$$

The expected pixel overhead is $s-1$. It increases monotonically with slice length s and surprisingly is independent of the length R of the "region-of-interest." Alas, the result is that simple only for 1-D.

If R itself is a random variable, then for a given value of $R = r$, (3) can be rewritten as

$$E\{P|R = r\} = r + s - 1. \tag{4}$$

b) *Analysis in 2-D:* We define two new random variables, P_w, the number of columns to be transmitted and P_h, the number of rows to be transmitted. Similarly, R_w and R_h are random variables denoting the number of columns and rows (among those transmitted) required to render the RoI respectively. From the 1-D analysis, we obtain

$$\begin{aligned} E\{P_w|R_w = r_w\} &= r_w + s_w - 1 \\ E\{P_h|R_h = r_h\} &= r_h + s_h - 1. \end{aligned}$$

The number of transmitted pixels is also a random variable, $P = P_w P_h$. Since P_w and P_h can be assumed to be conditionally independent given R_w, R_h, we can write

$$E\{P|R_w = r_w, R_h = r_h\} = (r_w + s_w - 1)(r_h + s_h - 1). \tag{5}$$

While $R_w \times R_h$ is the number of pixels among those transmitted which are rendered in the RoI window, it is *not* the size of the RoI window. The array of $R_w \times R_h$ pixels is resampled to fit the fixed size $d_w \times d_h$ of the RoI display window. Recall that this allows us to support arbitrary zoom factors with small number of discretely spaced resolution layers.

Random variable Z_C denotes the continuous zoom factor controlled by user input. Its value determines the value of the discrete random variable Z_D which is the zoom factor rounded to a power of two. For example

$$\begin{aligned} Z_D &= 1, \text{ if } (1 \leq Z_C < 1.5) \\ &\quad\ 2, \text{ if } (1.5 \leq Z_C < 4). \end{aligned} \tag{6}$$

To render the RoI at some zoom factor Z_C, we round to discrete zoom factor Z_D and retrieve the resolution layer $\log_2(Z_D)+1$. The mismatch Z_C/Z_D is made up by resizing the transmitted video after decoding. For our analysis, we need to model the conditional pdf of Z_C given the layer number. In our modeling below, we assume that, given the layer number, Z_C is uniformly distributed. For example, if the optimization is being carried out for the second layer in the example above, then we assume that Z_C is uniformly distributed between 1.5 and 4. Note that the distribution of the user-selected zoom factor in practice might depend on sizes of certain salient objects in the video. Nevertheless, we make the assumption about Z_C without performing any video content analysis.

Let d_w and d_h be constants denoting the width and height of the RoI display portion on the client's display, respectively. The random variables R_w and R_h are determined by Z_C as follows:

$$R_w = d_w \frac{Z_D}{Z_C} \quad R_h = d_h \frac{Z_D}{Z_C}. \tag{7}$$

The expected values of R_w and R_h are given by

$$\begin{aligned} E\{R_w\} &= d_w \times Z_D \times E\left\{\frac{1}{Z_C}\right\} \\ E\{R_h\} &= d_h \times Z_D \times E\left\{\frac{1}{Z_C}\right\} \end{aligned}$$

since the analysis is carried out given the layer number and hence the discrete zoom factor, Z_D. Now, we can apply iterated expectations on (5) to yield

$$E\{P\} = (E\{R_w\} + s_w - 1)(E\{R_h\} + s_h - 1). \tag{8}$$

2) *Optimal Slice Size:* The average number of bits per pixel for coding the prediction residual of a given resolution layer, denoted by $\eta(s_w, s_h)$, is a function of the slice size (s_w, s_h). We also define the number of pixels transmitted per rendered pixel as the relative pixel overhead $\psi(s_w, s_h) = \frac{E\{P\}}{d_w d_h}$, where $E\{P\}$ is given by (8). The optimal slice size minimizes the expected number of bits transmitted per rendered pixel and is given by

$$(s_w^{\text{opt}}, s_h^{\text{opt}}) = \arg\min_{(s_w, s_h)} \eta(s_w, s_h) \times \psi(s_w, s_h). \tag{9}$$

One way to obtain the function $\eta(s_w, s_h)$ is through sample encodings of the prediction residual by varying the slice

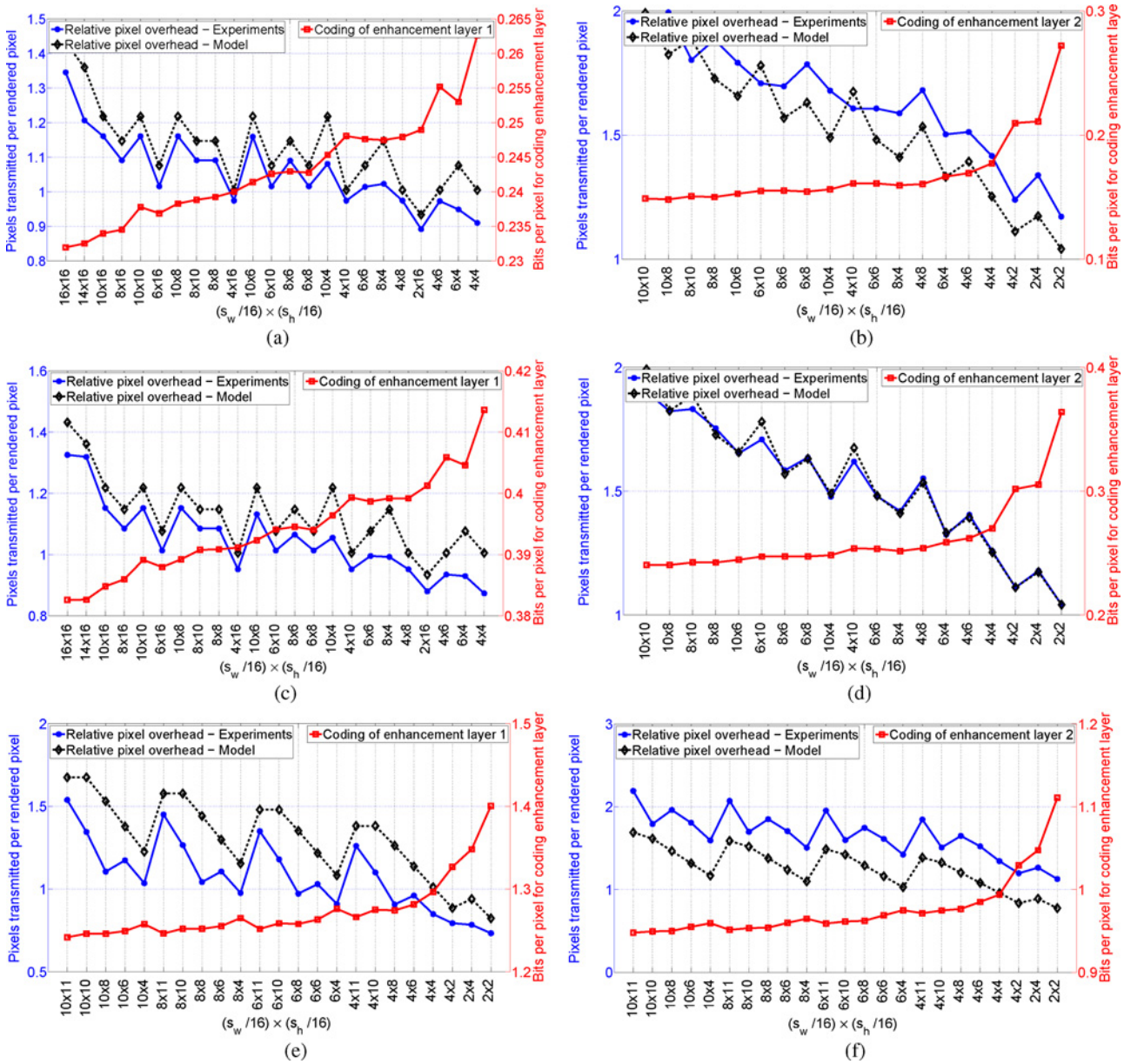

Fig. 5. Model prediction versus empirical values for pixels transmitted per rendered pixel, $\psi(s_w, s_h)$, shown for the three sequences, *Cardgame*, *Making Sense*, and *Soccer*. The empirical values are obtained by averaging over 100 user-interaction trajectories for each sequence. The second y-axis shows the bits per pixel for coding the residual of the high-resolution layer, $\eta(s_w, s_h)$. The slice width and slice height in number of pixels are denoted by s_w and s_h, respectively. (a) *Cardgame* sequence, layer 1 (PSNR $\approx$ 38.7 dB). (b) *Cardgame* sequence, layer 2 (PSNR $\approx$ 39.2 dB). (c) *Making Sense* sequence, layer 1 (PSNR $\approx$ 39.0 dB). (d) *Making Sense* sequence, layer 2 (PSNR $\approx$ 39.6 dB). (e) *Soccer* sequence, layer 1 (PSNR $\approx$ 35.5 dB). (f) *Soccer* sequence, layer 2 (PSNR $\approx$ 37.0 dB).

size. Alternatively, $\eta(s_w, s_h)$ could also be predicted by an analytical model to reduce the number of sample encodings. Either way, (9) can be used to find the optimal slice size.

We now present experimental results to demonstrate that our model predicts the optimal slice size accurately without requiring to capture user-interaction trajectories. In our experiments, we obtain $\eta(s_w, s_h)$ through a sample encoding of about 30 frames for each tested slice size configuration (s_w, s_h).

We use three video sequences for our experiments. The width $\times$ height of the *Cardgame*[2] and *Making Sense*[2] sequences is 3584 $\times$ 512 pixels. For the *Soccer*[3] sequence, it is 2560 $\times$ 704 pixels. The RoI display is 480 $\times$ 240 pixels. For all three sequences, the thumbnail video is obtained by spatially downsampling the original by 4 both horizontally and vertically. There are two high-resolution layers; the first layer sequence is obtained by downsampling the original by 2 both horizontally and vertically, while the second layer sequence is simply the original video. All sequences are 25 frames/s. *Cardgame* and *Making Sense* have 298 frames and *Soccer* has 598 frames. We encode the thumbnail videos with an

[2]Stanford Center for Innovations and Learning, Stanford, CA, generously provided these sequences.

[3]Fraunhofer Heinrich-Hertz Institute, Berlin, Germany, generously provided this sequence.

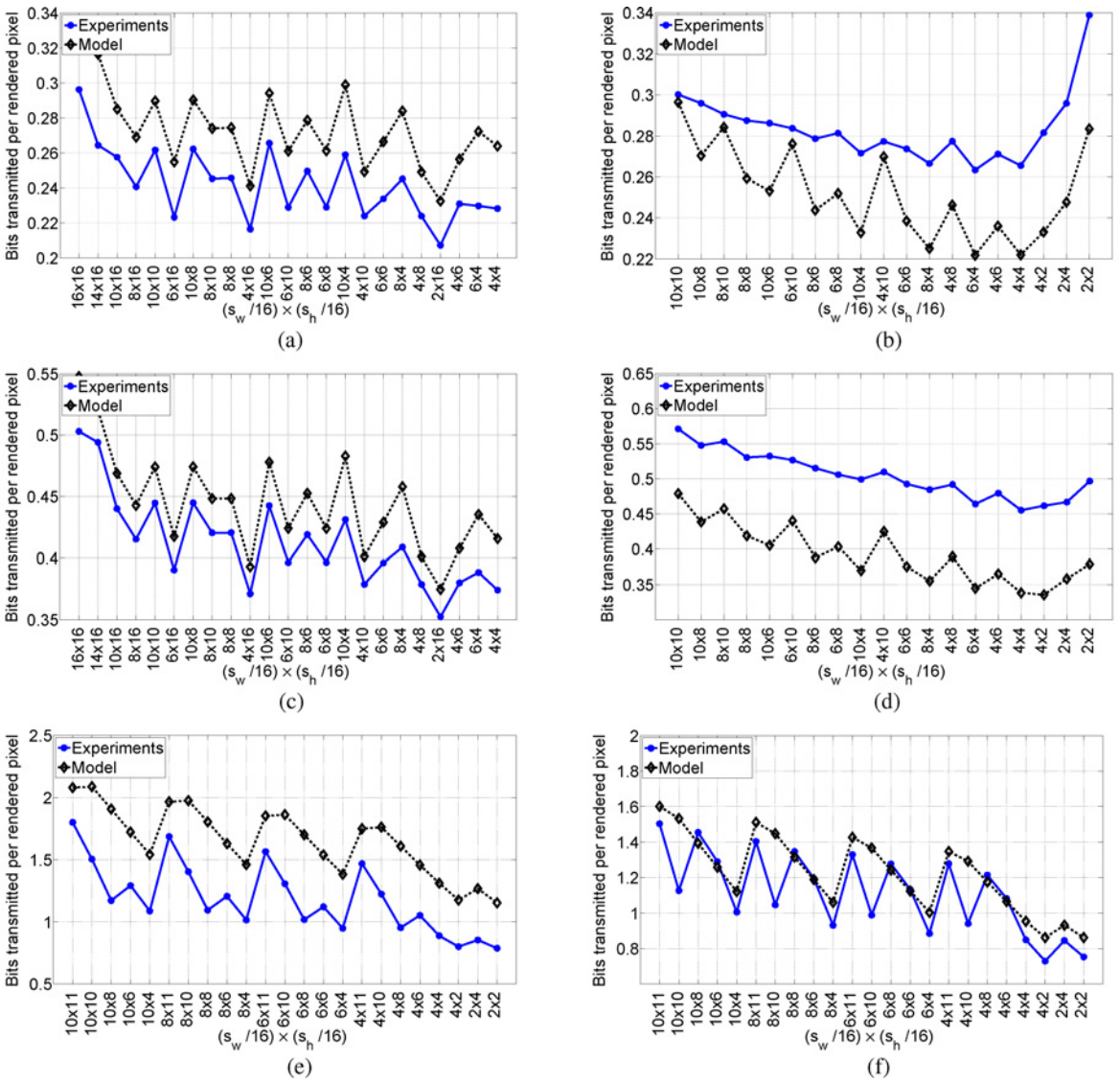

Fig. 6. Model prediction versus empirical values for bits transmitted per rendered pixel, shown for the three sequences, *Cardgame*, *Making Sense*, and *Soccer*. The empirical values are obtained by averaging over 100 user-interaction trajectories for each sequence. The slice width and slice height in number of pixels are denoted by s_w and s_h, respectively. (a) *Cardgame* sequence, layer 1 (PSNR $\approx$ 38.7 dB). (b) *Cardgame* sequence, layer 2 (PSNR $\approx$ 39.2 dB). (c) *Making Sense* sequence, layer 1 (PSNR $\approx$ 39.0 dB). (d) *Making Sense* sequence, layer 2 (PSNR $\approx$ 39.6 dB). (e) *Soccer* sequence, layer 1 (PSNR $\approx$ 35.5 dB). (f) *Soccer* sequence, layer 2 (PSNR $\approx$ 37.0 dB).

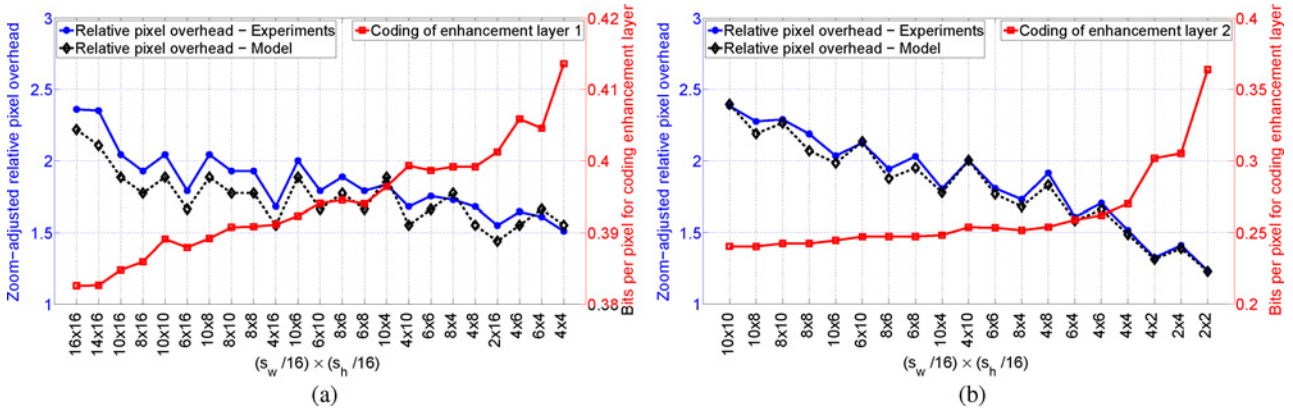

Fig. 7. Model prediction versus empirical values for zoom-adjusted relative pixel overhead, $\phi(s_w, s_h)$, shown for *Making Sense* sequence. The empirical values are obtained by averaging over 100 user-interaction trajectories. The second y-axis shows the bits per pixel for coding the residual of the high-resolution layer, $\eta(s_w, s_h)$. The slice width and slice height in number of pixels are denoted by s_w and s_h, respectively. (a) *Making Sense* sequence, layer 1 (PSNR $\approx$ 39.0 dB). (b) *Making Sense* sequence, layer 2 (PSNR $\approx$ 39.6 dB).

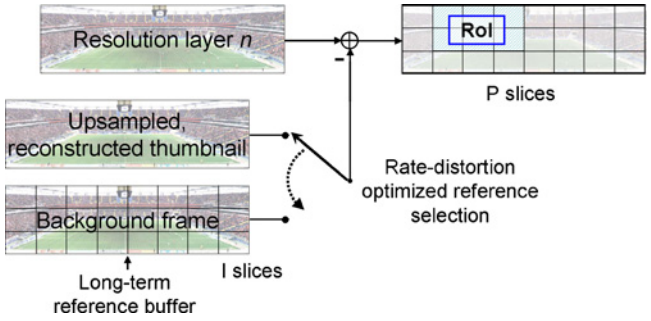

Fig. 8. Improvement based on background extraction. Each high-resolution layer frame has two references to choose from, the frame obtained by upsampling the reconstructed thumbnail frame and the background frame from the same layer in the background pyramid.

intraframe period of 15 frames using two consecutive B frames between anchor frames. The PSNR at bitrate for *Cardgame*, *Making Sense*, and *Soccer* is 39.1 dB at 162 kb/s, 39.6 dB at 201 kb/s, and 35.3 dB at 355 kb/s, respectively. For *Cardgame* and *Making Sense*, we choose the QPs to yield a PSNR of 38–40 dB for the high-resolution layers. For *Soccer*, the QPs yield a PSNR of 35.5–37 dB.

Fig. 5 shows the relative pixel overhead, $\psi(s_w, s_h)$ for the three sequences. We compare the model prediction against empirical values averaged over 100 user-interaction trajectories for each sequence. The trajectories were recorded while interactively viewing the sequence using the graphical user interface described in Section III. Each trajectory starts at a random location with a random zoom factor, is 1 min long, and the set of frames of the original sequence are looped to play for 1 min. The user's zoom factor, Z_C, is allowed to vary between 1 and 6. The thresholds given by (6) determine the high-resolution layer for rendering the RoI.

Fig. 6 shows the bits transmitted per rendered pixel for the three sequences. For a given sequence and resolution layer, the comparison in Figs. 5 and 6 for different slice sizes is made for the same QP and hence similar PSNR. Although the model predicts the optimal slice size fairly accurately, it can underestimate or overestimate the transmitted bitrate. This is because the popular slices that constitute the salient objects in the video could entail high or low bitrate compared to the average. Also, the location of the objects can bias the pixel overhead to the high or low side, whereas the model uses the average overhead. For certain zoom factors chosen by the user, the "accessed"/transmitted pixels could be less than the number of rendered pixels. This can be seen in Fig. 5 where the relative pixel overhead, $\psi(s_w, s_h)$, goes below one. Hence, we also compute the zoom-adjusted relative pixel overhead, $\phi(s_w, s_h) = E\left\{\frac{P_w P_h}{R_w R_h}\right\}$. This quantity is always greater than one

$$\phi(s_w, s_h) = \left[(s_w - 1)E\left\{\frac{1}{R_w}\right\} + 1\right]\left[(s_h - 1)E\left\{\frac{1}{R_h}\right\} + 1\right]$$

where

$$E\left\{\frac{1}{R_w}\right\} = \frac{E\{Z_C\}}{d_w Z_D} \quad E\left\{\frac{1}{R_h}\right\} = \frac{E\{Z_C\}}{d_h Z_D}.$$

Fig. 7 shows the zoom-adjusted relative pixel overhead, $\phi(s_w, s_h)$, for the *Making Sense* sequence. We observed that the model prediction is close to the empirical values for all three sequences. Thus, the analysis presented in this section enables estimating various quantities related to "accessed" portions from the scene representation without recording user-interaction trajectories and measuring these quantities from long bitstreams encoded for various slice sizes. This helps system dimensioning of an interactive video transmission system.

IV. Background Extraction and Long-Term Memory Motion-Compensated Prediction

The coding scheme proposed in Section III exploits temporal correlation by performing motion compensation among successive frames of the thumbnail video. Temporal prediction among successive frames of the high-resolution layers is avoided to enable efficient random access. Although it enables efficient random access, upward prediction using the reconstructed thumbnail frames might result in substantial residual energy for high spatial frequencies. In this section, we propose creating a background frame [27], [28] for each high-resolution layer and employing long-term memory motion-compensated prediction (LTM MCP) [29] to exploit the correlation between this frame and each high-resolution frame to be encoded. The background frame is intracoded. As shown in Fig. 8, high-resolution P slices have two references to choose from, upward prediction and the background frame. If a transmitted high-resolution P slice refers to the background frame, then relevant I slices from the background frame are transmitted only if they have not been transmitted earlier. This is different from [26], in which the encoder uses only those parts of the background for prediction that exist in the decoder's multi-resolution background pyramid. The encoder mimics the decoder in [26], which builds a background pyramid out of all previously received frames. Background extraction algorithms as well as detection and update of changed background portions have been previously studied, for example in [30], and are not the focus of this paper.

Since a moving camera might hamper spatial browsing experience, the camera is static in our sequences. A simple temporal median operator [27] yields a plausible background frame. Out of the first 150 frames, we include every fifth frame for the median operation. Fig. 9 shows the result for *Cardgame*, *Making Sense*, and *Soccer*. Although some stationary objects remain in the background frame, this helps the coding efficiency. In our experiments, the background frame is not updated after its creation at the start. This is typical with a static camera. For example, in a soccer game, the background typically changes due to illumination change, which happens infrequently. The background frame is intracoded with the same slice structure as the other frames from the layer. Fig. 10 shows the coding bitrate reduction due to this approach. The figure is shown for slice size of $\left(\frac{s_w}{16} \times \frac{s_h}{16}\right) = 4 \times 16$ for layer 1 and $\left(\frac{s_w}{16} \times \frac{s_h}{16}\right) = 6 \times 4$ for layer 2 of *Cardgame* and *Making Sense*. For *Soccer*, the slice size is $\left(\frac{s_w}{16} \times \frac{s_h}{16}\right) = 4 \times 4$ for both layers. For *Cardgame*, Fig. 11 shows the resulting transmission

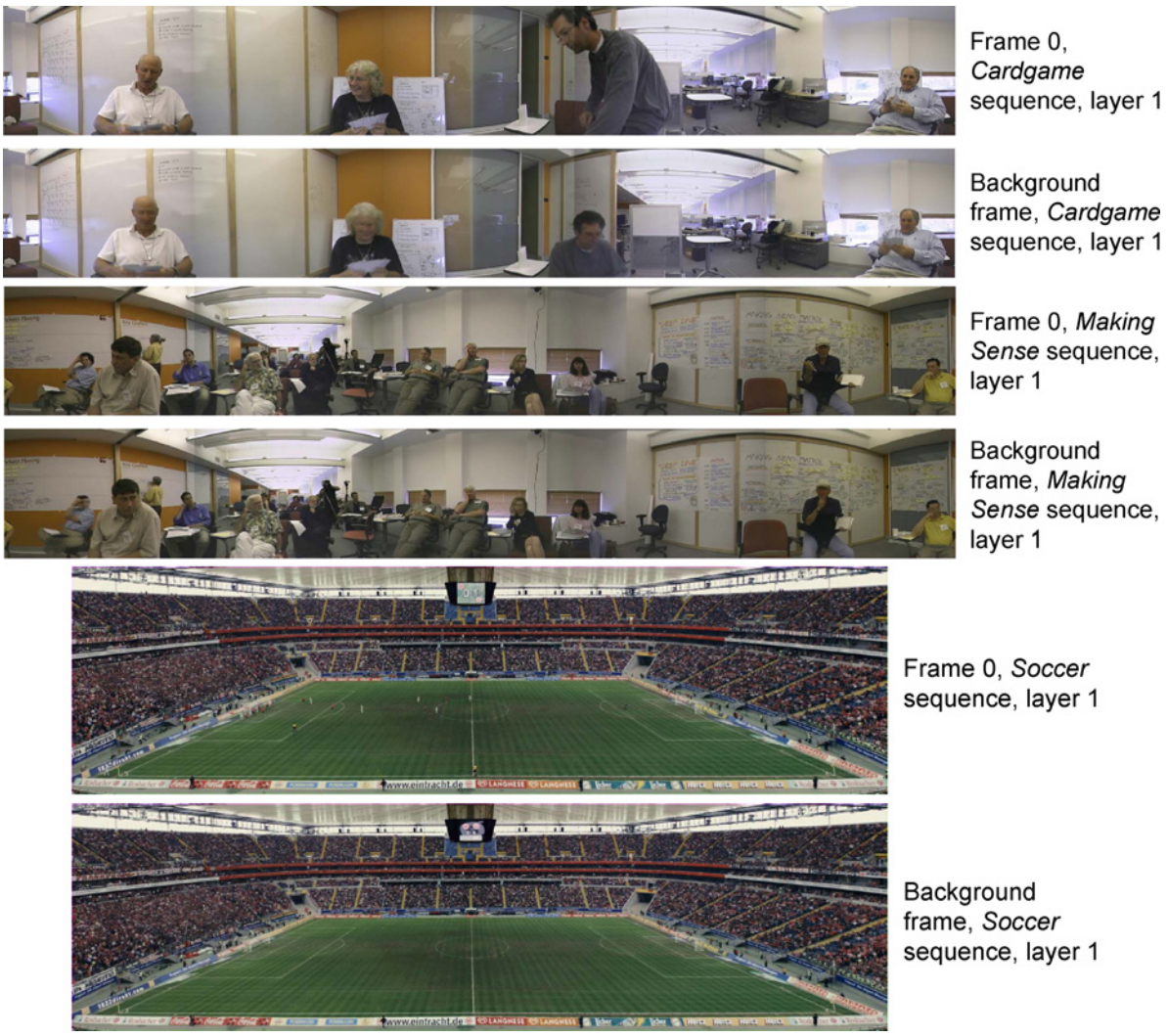

Fig. 9. Sample frame and background frame for layer 1 of *Cardgame*, *Making Sense*, and *Soccer* sequences.

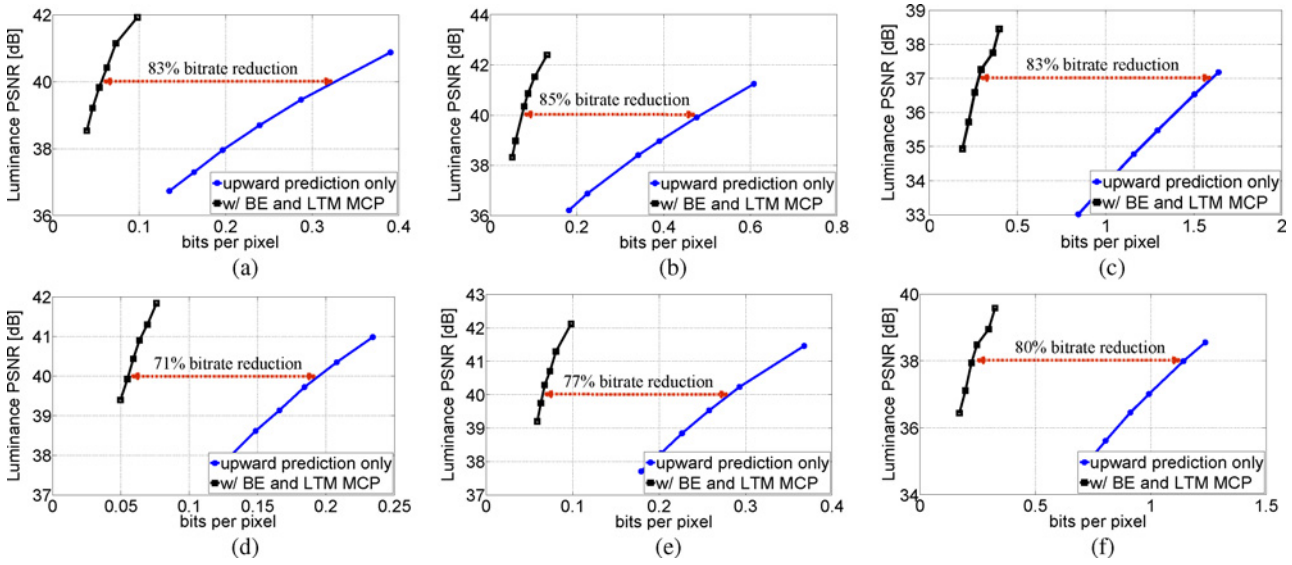

Fig. 10. Bitrate reduction through background extraction (BE) and long-term memory motion-compensated prediction (LTM MCP), shown for the *Cardgame*, *Making Sense*, and *Soccer* sequences. For both *Cardgame* and *Making Sense*, the slice size is $\left(\frac{s_w}{16} \times \frac{s_h}{16}\right) = 4 \times 16$ for layer 1 and $\left(\frac{s_w}{16} \times \frac{s_h}{16}\right) = 6 \times 4$ for layer 2. For *Soccer*, the slice size is $\left(\frac{s_w}{16} \times \frac{s_h}{16}\right) = 4 \times 4$ for both layers. (a) *Cardgame* sequence, layer 1. (b) *Making Sense* sequence, layer 1. (c) *Soccer* sequence, layer 1. (d) *Cardgame* sequence, layer 2. (e) *Making Sense* sequence, layer 2. (f) *Soccer* sequence, layer 2.

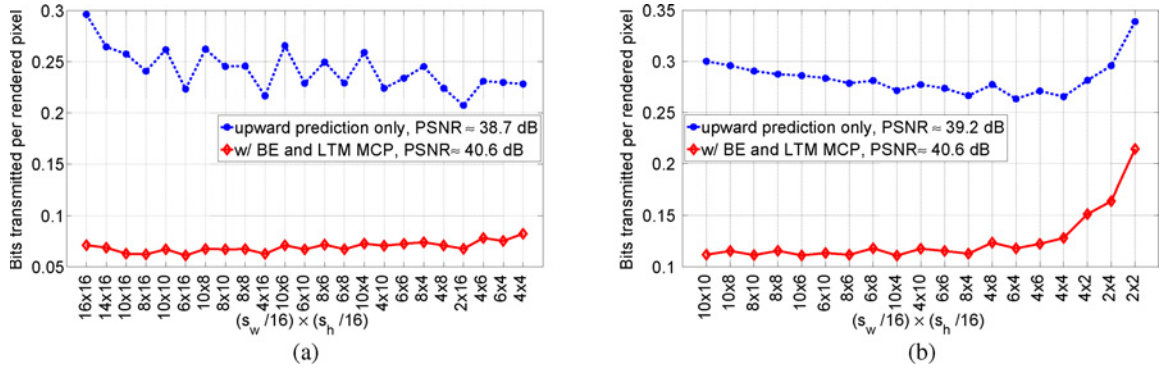

Fig. 11. Transmission bitrate is reduced after employing background extraction (BE) and long-term memory motion-compensated prediction (LTM MCP), here shown for the two layers of *Cardgame*. The slice width and slice height in number of pixels are denoted by s_w and s_h, respectively. Transmission bitrate values are obtained by counting bits required to transmit relevant high-resolution slices. The values are averaged over 100 user-interaction trajectories. (a) *Cardgame* sequence, layer 1. (b) *Cardgame* sequence, layer 2.

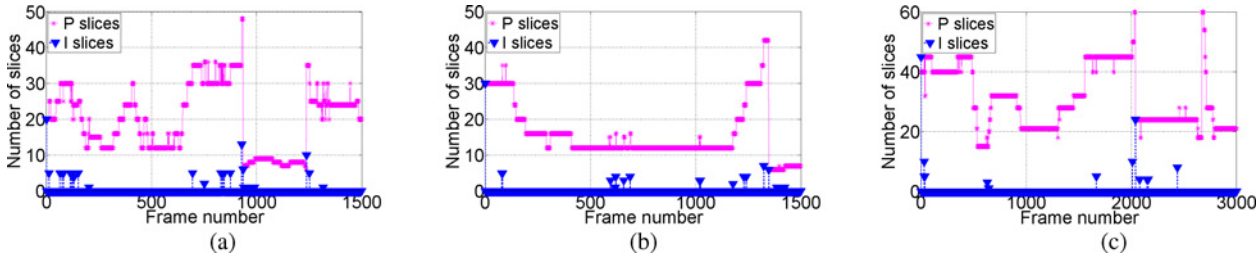

Fig. 12. Number of I and P slices transmitted over the streaming session, when background extraction (BE) and long-term memory motion-compensated prediction (LTM MCP) are employed. The data are plotted for a single user-interaction trajectory. Slice sizes are as in Fig. 10. For *Cardgame* and *Making Sense*, we choose the QPs to yield around 40.6 dB PSNR for both layers. For *Soccer*, the PSNR is around 37.3 dB for layer 1 and 38.5 dB for layer 2. (a) *Cardgame* sequence. (b) *Making Sense* sequence. (c) *Soccer* sequence.

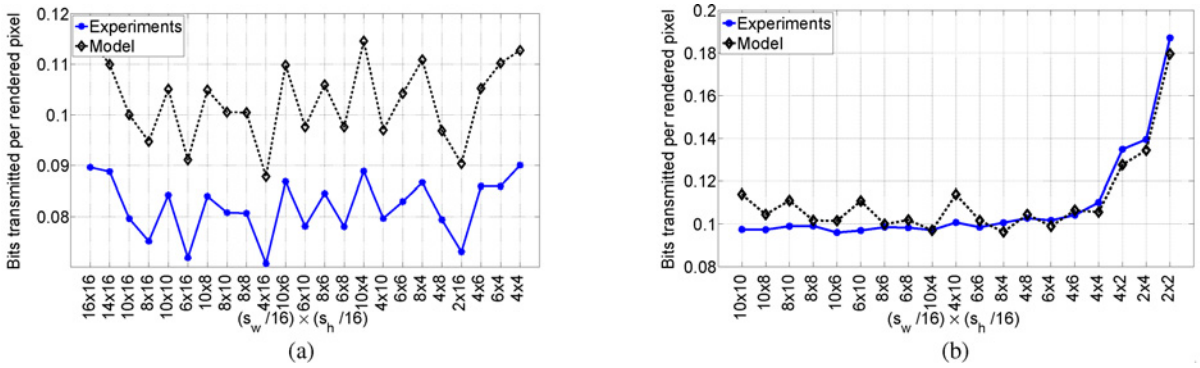

Fig. 13. Model prediction versus empirical values for bits transmitted per rendered pixel, shown for the *Making Sense* sequence, encoded using background extraction (BE) and long-term memory motion-compensated prediction (LTM MCP). The empirical values are obtained by averaging over 100 user-interaction trajectories. The slice width and slice height in number of pixels are denoted by s_w and s_h, respectively. (a) *Making Sense* sequence, layer 1 (PSNR $\approx$ 40.6 dB). (b) *Making Sense* sequence, layer 2 (PSNR $\approx$ 40.6 dB).

bitrate reduction. For Fig. 10, the slice sizes chosen are either optimal or close to optimal. If the mean transmission bitrates corresponding to two slice sizes are close, we prefer the larger slice size for reasons noted in Section V.

For the high-resolution layers, Fig. 12 shows the number of transmitted I slices from the background pyramid and the number of transmitted P slices. It shows the numbers for a single user-interaction trajectory. For the first frame of the streaming session, roughly equal numbers of I and P slices are transmitted. Subsequently, I slices need to be transmitted sporadically in time and generally fewer in number than at the start. Although not shown here, when averaged over 100 trajectories, the profiles of the transmitted I and P slices appear smoother; the number of P slices is almost constant and matches the expected number of transmitted P slices that can be computed from analysis similar to Section III-C. The average number of transmitted I slices is highest at the start and is about 1% of the number of transmitted P slices thereafter.

We model the bits transmitted per rendered pixel as before. However, for simplicity, the cost of transmitting I slices is counted in the coding bitrate, $\eta(s_w, s_h)$, but not in the number of pixels transmitted per rendered pixel, $\psi(s_w, s_h)$. As shown in Fig. 13, the model matches closely with the empirical

values for the *Making Sense* sequence. The model matches well for the other two sequences as well. It should be noted that the change in the optimal slice size after employing the background frame is small, and the slice size that is optimal for the earlier scheme still yields a mean transmission bitrate very close to that corresponding to the new optimal slice size. Hence, we choose the same slice size for comparing the coding bitrates of the two schemes in Fig. 10.

V. Conclusion and Further Work

We proposed a spatial-random-access-enabled video coding scheme that eliminates the need to transmit and/or decode the entire video scene in high spatial resolution. The RoI can be switched during any frame interval without waiting for the end of the GoP or having to transmit extra slices from the past. The coding scheme allows the system to scale with the number of clients; it avoids encoding each client's RoI sequence individually. Another benefit is that requested RoIs can be extracted from the bitstream even inside or at the edge of the network, closer to the client-nodes. The random access aspects presented in this paper also apply to the design of other IBR-based interactive streaming systems.

We optimized the slice size to minimize the transmission bitrate. Our model accurately predicts the optimal slice size without requiring to capture user-interaction trajectories. We proposed an improvement of the coding scheme based on background extraction and long-term memory motion-compensated prediction. Experiments indicate that both the coding bitrate as well as the transmission bitrate can be reduced by up to 85% while retaining efficient random access capability. This improvement, however, entails transmitting some I slices from the background pyramid that might be required for decoding the current high-resolution P slice. Nevertheless, the cost of doing this is amortized over the streaming session.

For reducing latency in a streaming scenario, we proposed predicting the user's RoI in advance [31], [32] and pre-fetching relevant data. A bigger slice size adds robustness against inaccurate RoI prediction, although it might increase transmission bitrate. Also, if the packetization overhead associated with layers below the application layer is considered, for example when each slice needs to be put in a different transport layer packet, then a bigger slice size might be optimal. A sample scenario is application-layer P2P multicast to a population of peers where each peer can subscribe/unsubscribe requisite tiles according to its RoI. In [33] and [34], we proposed forming a multicast group for each slice. In this scenario, data from disjoint slices are preferably transmitted/forwarded in different transport layer packets. In the RoI P2P system, the peer's task of deciding which multicast groups to subscribe is simplified thanks to efficient random access of the underlying video coding scheme.

Acknowledgment

The authors would like to thank Dr. P. Baccichet, Dr. D. Varodayan, and K. Chono for useful discussions.

References

[1] C. Fehn, C. Weissig, I. Feldmann, M. Mueller, P. Eisert, P. Kauff, and H. Bloss, "Creation of high-resolution video panoramas of sport events," in *Proc. 8th IEEE ISM*, Dec. 2006, pp. 291–298.

[2] J. Kopf, M. Uyttendaele, O. Deussen, and M. F. Cohen, "Capturing and viewing gigapixel images," in *Proc. ACM SIGGRAPH*, vol. 26, no. 3. Aug. 2007, pp. 93.1–93.10.

[3] Hewlett-Packard. (2009, Sep. 16). *Halo: Video Conferencing Product by Hewlett-Packard* [Online]. Available: http://www.hp.com/halo/index.html

[4] A. Smolic and D. McCutchen, "3DAV exploration of video-based rendering technology in MPEG," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 3, pp. 348–356, Mar. 2004.

[5] Immersive Media. (2009, Sep. 16). *Dodeca 2360: An Omni-Directional Video Camera Providing Over 100 Million Pixels per Second by Immersive Media* [Online]. Available: http://www.immersivemedia.com

[6] *Video Clip Showcasing Interactive TV with Pan/Tilt/Zoom* (2009, Sep. 26) [Online]. Available: http://www.youtube.com/watch?v=Ko9jcIjBXnk

[7] H.-Y. Shum, S. B. Kang, and S.-C. Chan, "Survey of image-based representations and compression techniques," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 11, pp. 1020–1037, Nov. 2003.

[8] M. Levoy and P. Hanrahan, "Light field rendering," in *Proc. ACM SIGGRAPH*, Aug. 1996, pp. 31–42.

[9] P. Kauff and O. Schreer, "Virtual team user environments: A step from tele-cubicles toward distributed tele-collaboration in mediated workspaces," in *Proc. IEEE ICME*, vol. 2. Aug. 2002, pp. 9–12.

[10] M. Tanimoto, "Overview of FTV (free-viewpoint television)," in *Proc. ICME*, Jul. 2009, pp. 1552–1553.

[11] D. Taubman and R. Rosenbaum, "Rate-distortion optimized interactive browsing of JPEG2000 images," in *Proc. IEEE ICIP*, Sep. 2000, pp. 765–768.

[12] D. Taubman and R. Prandolini, "Architecture, philosophy and performance of JPIP: Internet protocol standard for JPEG2000," *Proc. SPIE Intl. Symp. VCIP*, vol. 5150, no. 1, pp. 791–805, Jul. 2003.

[13] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 390–404, Apr. 2006.

[14] B. Girod, "The efficiency of motion-compensating prediction for hybrid coding of video sequences," *IEEE J. Sel. Areas Commun.*, vol. 5, no. 7, pp. 1140–1154, Aug. 1987.

[15] B. Girod, "Motion-compensating prediction with fractional-PEL accuracy," *IEEE Trans. Commun.*, vol. 41, no. 4, pp. 604–612, Apr. 1993.

[16] B. Girod, "Efficiency analysis of multihypothesis motion-compensated prediction for video coding," *IEEE Trans. Image Process.*, vol. 9, no. 2, pp. 173–183, Feb. 2000.

[17] S. Heymann, A. Smolic, K. Mueller, Y. Guo, J. Rurainsky, P. Eisert, and T. Wiegand, "Representation, coding and interactive rendering of high-resolution panoramic images and video using MPEG-4," in *Proc. PPW*, Feb. 2005.

[18] P. Ramanathan and B. Girod, "Rate-distortion optimized streaming of compressed light fields with multiple representations," in *Proc. 14th Packet Video Workshop*, Dec. 2004.

[19] P. Ramanathan and B. Girod, "Random access for compressed light fields using multiple representations," in *Proc. IEEE 6th Int. Workshop MMSP*, Sep. 2004, pp. 383–386.

[20] M. Karczewicz and R. Kurceren, "The SP- and SI-frames design for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 637–644, Jul. 2003.

[21] X. Zhu, A. Aaron, and B. Girod, "Distributed compression for large camera arrays," in *Proc. IEEE Workshop Statist. Signal Process.*, Sep. 2003, pp. 30–33.

[22] A. Aaron, P. Ramanathan, and B. Girod, "Wyner–Ziv coding of light fields for random access," in *Proc. IEEE 6th Workshop MMSP*, Sep. 2004, pp. 323–326.

[23] I. Bauermann and E. Steinbach, "RDTC optimized compression of image-based scene representations (part I): Modeling and theoretical analysis," *IEEE Trans. Image Process.*, vol. 17, no. 5, pp. 709–723, May 2008.

[24] I. Bauermann and E. Steinbach, "RDTC optimized compression of image-based scene representations (part II): Practical coding," *IEEE Trans. Image Process.*, vol. 17, no. 5, pp. 724–736, May 2008.

[25] E. Kurutepe, M. R. Civanlar, and A. M. Tekalp, "Interactive transport of multi-view videos for 3DTV applications," *J. Zhejiang Univ. Sci. A*, vol. 7, no. 5, pp. 830–836, May 2006.

[26] J. Bernstein, B. Girod, and X. Yuan, "Hierarchical encoding method and apparatus employing background references for effi-

ciently communicating image sequences," U.S. Patent 5 155 594, Oct. 1992.
[27] M. Massey and W. Bender, "Salient stills: Process and practice," *IBM Syst. J.*, vol. 35, nos. 3–4, pp. 557–573, 1996.
[28] D. Farin, P. de With, and W. Effelsberg, "Robust background estimation for complex video sequences," in *Proc. IEEE ICIP*, vol. 1. Sep. 2003, pp. 145–148.
[29] T. Wiegand, X. Zhang, and B. Girod, "Long-term memory motion-compensated prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 1, pp. 70–84, Feb. 1999.
[30] D. Hepper, "Efficiency analysis and application of uncovered background prediction in a low bit rate image coder," *IEEE Trans. Commun.*, vol. 38, no. 9, pp. 1578–1584, Sep. 1990.
[31] A. Mavlankar, D. Varodayan, and B. Girod, "Region-of-interest prediction for interactively streaming regions of high resolution video," in *Proc. IEEE 16th Packet Video Workshop*, Nov. 2007, pp. 68–77.
[32] A. Mavlankar and B. Girod, "Pre-fetching based on video analysis for interactive region-of-interest streaming of soccer sequences," in *Proc. IEEE ICIP*, Nov. 2009, pp. 3061–3064.
[33] A. Mavlankar, J. Noh, P. Baccichet, and B. Girod, "Peer-to-peer multicast live video streaming with interactive virtual pan/tilt/zoom functionality," in *Proc. IEEE ICIP*, Oct. 2008, pp. 2296–2299.
[34] A. Mavlankar, J. Noh, P. Baccichet, and B. Girod, "Optimal server bandwidth allocation for streaming multiple streams via P2P multicast," in *Proc. IEEE 10th Workshop MMSP*, Oct. 2008, pp. 468–473.

Aditya Mavlankar (S'99–M'09) received the B.E. degree in electronics and telecommunication from the University of Pune, Pune, India, the M.S. degree in communications engineering from the Technical University of Munich, Munich, Germany, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA.

He is currently with Tely Labs, Inc., Menlo Park, CA. He has published over 30 conference and journal papers, book chapters, and patents. His current research interests include scalable video coding, interactive video delivery, and peer-to-peer video streaming.

Dr. Mavlankar was the recipient of the Edison Prize Bronze Medal awarded by IIE Europe in conjunction with the GE Foundation for his Masters thesis in 2006, was a co-recipient of the Best Student Paper Award at the IEEE Workshop on Multimedia Signal Processing, Victoria, BC, Canada, and a co-recipient of the Best Student Paper Award at the European Signal Processing Conference, Poznan, Poland. He won the Student Travel Grant Award for his paper at the 16th International Packet Video Workshop, Lausanne, Switzerland. Papers co-authored by him have been nominated multiple times for best paper awards at international conferences.

Bernd Girod (M'80–SM'97–F'98) received the M.S. degree from the Georgia Institute of Technology, Atlanta, and the Engineering Doctorate degree from the University of Hannover, Hannover, Germany.

He has been a Professor of electrical engineering and (by courtesy) computer science with the Information Systems Laboratory, Stanford University, Stanford, CA, since 1999. Previously, he was a Professor of telecommunications with the Department of Electrical Engineering, University of Erlangen-Nuremberg, Erlangen/Nuremberg, Germany. He has published over 400 conference and journal papers, as well as 5 books. His current research interests include the areas of video compression and networked media systems.

Prof. Girod received the EURASIP Signal Processing Best Paper Award in 2002, the IEEE Multimedia Communication Best Paper Award in 2007, the EURASIP Image Communication Best Paper Award in 2008, as well as the EURASIP Technical Achievement Award in 2004. As an entrepreneur, he has been involved with several startup ventures as the founder, director, investor, or advisor, among them Polycom (Nasdaq:PLCM), Vivo Software, 8x8 (Nasdaq: EGHT), and RealNetworks (Nasdaq: RNWK). He is a EURASIP fellow and a member of the German National Academy of Sciences (Leopoldina).