Video Streaming with Interactive Pan/Tilt/Zoom

Aditya Mavlankar and Bernd Girod

Abstract High-spatial-resolution videos offer the possibility of viewing an arbitrary region-of-interest (RoI) interactively. The user can pan/tilt/zoom while watching the video. This chapter presents spatial-random-access-enabled video compression that encodes the content such that arbitrary RoIs corresponding to different zoom factors can be extracted from the compressed bit-stream. The chapter also covers RoI trajectory prediction, which allows pre-fetching relevant content in a streaming scenario. The more accurate the prediction the lower is the percentage of missing pixels. RoI prediction techniques can perform better by adapting according to the video content in addition to simply extrapolating previous moves of the input device. Finally, the chapter presents a streaming system that employs application-layer peer-to-peer (P2P) multicast while still allowing the users to freely choose individual RoIs. The P2P overlay adapts on-the-fly for exploiting the commonalities in the peers' RoIs. This enables peers to relay data to each other in real-time, thus drastically reducing the bandwidth required from dedicated servers.

1 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 [24, 36]. Some currently available video-conferencing systems stitch

Aditya Mavlankar
Stanford University, 350 Serra Mall, Stanford CA 94305, USA, e-mail: maditya@stanford.edu

Bernd Girod
Stanford University, 350 Serra Mall, Stanford CA 94305, USA, e-mail: bgirod@stanford.edu

a large panoramic view in real-time [2]. 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 [67]. An example of such an acquisition device is [1].

Imagine that a user wants to watch a high-spatial-resolution video that exceeds the resolution of his/her display screen. If the user were to watch a downsampled version of the video that fits the display screen then he/she might not be able to view local regions with the recorded high resolution. A possible solution to this problem is a video player that supports interactive pan/tilt/zoom. The user can thus choose to watch an arbitrary region-of-interest (RoI). We refer to this functionality as interactive region-of-interest (IRoI). Figure 1 shows screen-shots of a video player supporting IRoI. Such a video player could also offer to track certain objects, whereby the user is not required to control pan and tilt, but could still control the zoom factor.

Some practical scenarios where this kind of interactivity is well-suited are: interactive playback of high-resolution video from locally stored media, 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., lectures, panel discussions). A video clip that showcases interactive viewing of soccer in a TV-like setting can be seen here [3].

Consider the first example mentioned above, i.e., playback from locally stored media. In this case, the video content is encoded offline before storing it on the relevant media, for example, a high-capacity portable disk. Note that the RoI trajectory is not known while encoding the content. An RoI trajectory is determined each time a user watches the video with interactive pan/tilt/zoom. This leads us to two design choices; 1) the video player can be designed to decode the entire high spatial resolution while displaying only the RoI or 2) the adopted compression format could allow decoding only relevant regions, possibly with some overhead. Depending on the resolution of the video and the hardware capability of the player, the first design choice might be prohibitive. Other application scenarios mentioned above entail streaming from a remote source. In most cases, streaming the full spatial extent of the video to a user can be ruled out due to prohibitive bandwidth requirement. If RoI-specific portions can be streamed to the remote user, the RoI dimensions could be adapted to suit the available data rate for communication apart from the user's display screen as noted above.

Now let us consider the difficulty of employing a standard video encoder in the streaming scenario. A standard video encoder generally does not provide efficient spatial random access, i.e., the ability to extract regions from the compressed bitstream. The video streaming can be for live content or for pre-stored content. For live content, the server can crop out an RoI sequence on-the-fly considering the user's pan/tilt/zoom commands and compress it as a video sequence using standard video encoding. The load of encoding might get prohibitively large with increasing numbers of users. Pre-stored content might not be stored in raw format implying that the server has to decode the high-spatial-resolution video prior to cropping the RoI se-

quence. Not only does the load of encoding increase, but if multiple users watch the content asynchronously then even the decoding load at the server increases. On the other hand, if a spatial-random-access-enabled video coding scheme is employed, the server needs to encode the recorded field-of-view only once, possibly with multiple resolution layers to support different zoom factors. The encoding load can thus

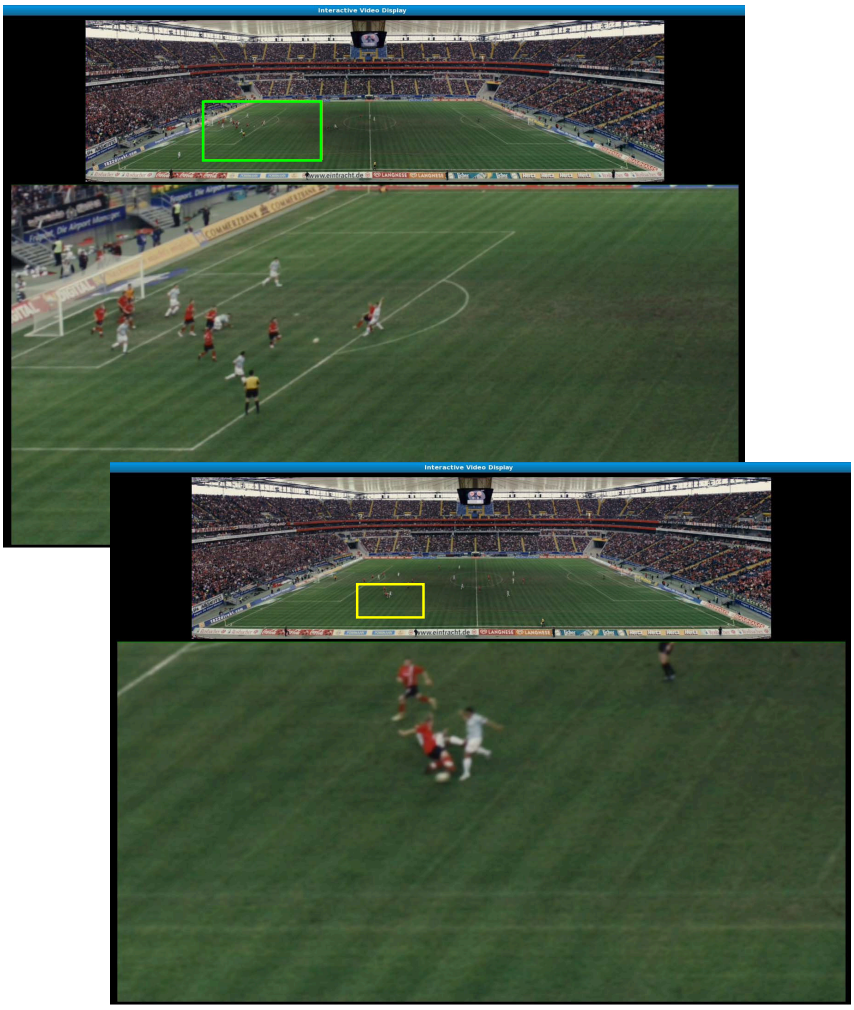

Fig. 1 Screen-shots of a video player supporting interactive pan/tilt/zoom. Apart from displaying the RoI, the video player can display a thumbnail overview to aid navigation in the scene. The player could also offer to track certain objects, for example, the soccer ball and/or the soccer players. In the tracking mode, the user is not required to control pan and tilt, but could still control the zoom factor.

be upper-bounded both for live content as well as pre-stored content irrespective of the number of users.

In addition to limiting the encoding load, if the streaming bandwidth required from the server can also be limited then the streaming system can scale to large numbers of users. This chapter presents a solution that can be employed when several users are synchronously watching arbitrary regions of a high-spatial-resolution video. It hinges on employing application-layer peer-to-peer (P2P) multicast for delivering the streams to the users. The solution exploits the commonalities in the peers' regions such that they relay data to each other in real-time. This allows limiting the bandwidth required at the server by making use of the forwarding capacities of the peers. The main challenge is that user-interaction determines real-time which regions are commonly wanted by which peers. The P2P overlay needs to adapt quickly and in a distributed manner, i.e., peers take most of the action necessary for acquiring the data they need, without much central intervention. Larger dependence on central intervention represents another hurdle in scaling. The second challenge is that peers can switch off randomly, taking away the resources they bring with them.

Ideally, the changing RoI should be rendered immediately upon user input; i.e., without waiting for new data to arrive. If the client would delay the rendering until new data arrive, the induced latency might hamper the experience of interactivity. In both client-server unicast streaming as well as P2P multicast streaming, predicting the user's navigation path ahead of time helps pre-fetch relevant sub-streams. The more accurate the RoI prediction the lower is the percentage of pixels that have to be error-concealed.

This chapter is organized as follows. Section 2 provides a sampling of interactive streaming systems found in the literature. The goal is to highlight the challenges as well as earlier proposed approaches for providing random access, enabling pre-fetching and P2P design for other interactive applications that are similar in spirit to IRoI video. Section 3 discusses several approaches for providing spatial random access within videos. It elaborates one video coding scheme in particular. This scheme builds a multi-resolution pyramid comprising slices. It is shown how background extraction can be used to improve the coding efficiency of such a scheme. The trade-off in choosing the slice size is also analyzed. The slice size can be chosen to strike the right balance between storage requirement and transmission bit-rate. Section 4 describes variants of pre-fetching schemes. In one of the variants, the RoI prediction is based on analyzing the motion of objects in the video in addition to extrapolating moves of the input device. RoI prediction can be carried out at the client, at the server or collectively. Section 5 presents the P2P multicasting system in which peers can control their individual RoIs. Key aspects of the design are presented that enable peers to receive and relay respective regions despite the challenges outlined above.

2 Related Work

This section draws on interactive streaming systems found in the literature. A brief survey of the challenges in designing such systems and the solutions found in the literature sets the stage for the discussion on video streaming with interactive pan/tilt/zoom appearing in later sections. The later sections particularly aim at building a system that scales to large numbers of users.

2.1 Coding for Random Access

Images Remote image browsing with interactive pan/tilt/zoom is very similar in spirit. It is generally used for high-resolution archaeological images, aerial or satellite images, images of museum exhibits, online maps, etc. Online maps provide about 20 zoom levels. The image corresponding to each zoom level is coded into tiles. Generally, the images corresponding to different zoom levels are coded independently. This so-called Gaussian pyramid fails to exploit redundancy across zoom levels but provides easy random access. The server accesses the tiles intersecting the selected view and sends these tiles to the user. Generally, after a zoom operation, the relevant part from the current zoom level is interpolated to quickly render the newly desired view. As the tiles from the new zoom level arrive, the graphics become crisper. Note that this cursory rendering based on earlier received data might not be possible for some portions due to lack of received data.

Interactive browsing of images using JPEG2000 is explored in [5, 72]. This leverages the multi-resolution representation of an image using wavelets. This representation is not overcomplete unlike the Gaussian and Laplacian pyramids that generate more coefficients than the high-resolution image. JPEG2000 encodes blocks of wavelet transform coefficients independently. This means that 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 sub-bitstream. This makes it possible to stream any given block with a desired degree of fidelity. A transmission protocol, called JPEG2000 over Internet Protocol (JPIP), has also been developed. The protocol governs communication between a client and a server to support remote interactive browsing of JPEG2000 coded images [71]. The server can keep track of the RoI trajectory of the client as well as the parts of the bit-stream 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 bit-stream need to be sent in order to maximize the quality of the current RoI.

Video The video compression standard H.264/AVC [4, 74] includes tools like Flexible Macroblock Ordering (FMO) and Arbitrary Slice Ordering (ASO). These tools were primarily created for error resilience, but can also be used to define an RoI prior to encoding [21]. The RoI can either be defined through manual input or through au-

tomatic content analysis. Slices corresponding to the RoI (or multiple RoIs) can be encoded with higher quality compared to other regions. Optionally, the scalable extension of H.264/AVC, called SVC [6, 59], can be used for adding fine or coarse granular fidelity refinements for RoI slices. The user experiences higher quality for the RoI if the refinement packets are received. The RoI encoding parameters can be adapted to the network and/or the user [12]. Note that these systems transmit the entire picture while delivering the RoI with higher quality. Among the class of such systems, some employ JPEG2000 with RoI support and conditional replenishment for exploiting correlation among successive frames [20]. Parts of the image that are not replenished can be copied from the previous frame or a background store.

In our own work, we have proposed a video transmission system for interactive pan/tilt/zoom [44]. This system crops the RoI sequence from the high-resolution video and encodes it using H.264/AVC. The RoI cropping is adapted to yield efficient motion compensation in the video encoder. The RoI adjustment is confined to ensure that the user does not notice the manipulation and experiences accurate RoI control. The normal mode of operation for this system is streaming live content but we also allow the user to rewind and play back older video. Note that in the second mode of operation, the high-resolution video is decoded prior to cropping the RoI sequence. Although efficient in terms of transmitted bit-rate, the drawback is that RoI video encoding has to be invoked for each user, thus limiting the system to few users. This system targets remote surveillance in which the number of simultaneous users is likely to be less than other applications like interactive TV.

Video coding for spatial random access presents a special challenge. To achieve good compression efficiency, video compression schemes typically exploit correlation among successive frames. This is accomplished through motion-compensated interframe prediction [26, 27, 28]. However, this makes it difficult to provide random access for spatial browsing within the scene. This is because 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 and/or decoded earlier.

Coding, transmission and rendering of high-resolution panoramic videos using MPEG-4 is proposed in [29, 31]. A limited part of the entire scene is transmitted to the client depending on the chosen viewpoint. Only intraframe coding is used to allow random access. The scene is coded into independent slices. The authors mention the possibility of employing interframe coding to gain more compression efficiency. However, they note that this involves transmitting slices from the past if the current slice requires those for its decoding. A longer intraframe period entails significant complexity for slices from the latter frames in the group of pictures (GOP), as this "dependency chain" grows.

Multi-View Images/Videos Interactive streaming systems that provide virtual fly-around in the scene employ novel-view generation to render views of the scene from arbitrary viewpoints. With these systems, the user can experience more free interactive navigation in the scene [33, 70, 68]. These systems typically employ image-based rendering (IBR) which is a technique to generate the novel view from

multiple views of the scene recorded using multiple cameras [65, 41]. Note that in these applications, the scene itself might or might not be evolving in time. Transmitting arbitrary views from the multi-view data-set on-the-fly also entails random access issues similar to those arising for transmitting arbitrary regions in interactive pan/tilt/zoom. Interframe coding for compressing successive images in time as well as from neighboring views can achieve higher compression efficiency but can lead to undesirable dependencies for accessing random views. There exists a large body of works that employs hybrid video coding for compressing multi-view data-sets [13, 14, 34, 42, 25]. These studies highlight the trade-off in storage requirement, mean transmission bit-rate and decoding complexity. Recently, an analytical framework was proposed for optimizing the coding structure for coding multi-view data-sets [16]. The framework allows multiple representations of a picture, for example, compressed using different reference pictures. The optimization not only finds the best coding structure but also determines the best set of coded pictures to transmit corresponding to a viewing path. The framework can accommodate constraints like limited step-size for view switching, permitting view switching only during certain frame-intervals and capping the length of the burst of reference frames that are used for decoding a viewed frame but are not themselves displayed. The framework can minimize a weighted sum of expected transmission bit-rate and storage cost for storing the compressed pictures.

The video compression standard H.264/AVC defines two new slice types, called SP and SI slices. Using these slice types, it is possible to create multiple representations of a video frame using different reference frames. Similar to the solutions described above, the representation to be streamed is chosen according to the reference frames available at the decoder. However, the novelty is that the reconstruction is guaranteed to be identical. This drastically reduces the total number of multiple representations required to be stored. SP frames have been used for interactive streaming of static light fields [57, 56]. Another solution to the random access problem associated with multi-view data-sets is based on distributed source coding (DSC) [32, 7]. In this solution, an interframe coded picture is represented using enough parity bits which leads to an identical reconstruction irrespective of the reference frame used by the decoder. This implies that multiple representations are not required to be stored, however, the number of parity bits is determined by the reference frame having the least correlation to the frame to be coded. Similar to some prior work based on hybrid video coding for multi-view data-sets mentioned above, recent work based on DSC also explores the trade-off between transmission bit-rate and storage requirement [17].

2.2 Navigation Path Prediction

A simple user-input device, for example a computer mouse, typically senses position. More sophisticated devices like game-controllers can also measure velocity and/or acceleration. Studies on view trajectory prediction have been conducted in

the context of Virtual Reality [10] and networked multi-player video games [66]. A common navigation path prediction technique, dead reckoning, predicts the future path by assuming that the user maintains the current velocity. The velocity can be either read from the input device or computed from successive position measurements.

In their work on interactive streaming of light fields, the authors predict the (x, y) mouse co-ordinates based on dead-reckoning and translate these into the viewpoint [58]. The use of a Kalman filter for head movement prediction in scenarios where head movements can control the application have been proposed in [35]. In prior work on dynamic light fields, six Kalman filters have been used for predicting the 3-D co-ordinates and the 3 Euler angles that define the viewpoint [37, 38]. The viewpoint and the rendering algorithm together determine the number of views that need to be streamed to the client. The authors mention two possible system design choices. Viewpoints exceeding the bit-rate threshold can be disallowed or those viewpoints can be rendered with lower quality by not streaming all the views demanded by that viewpoint. The authors also note that if the streaming system allows tuning into a view-stream only during certain frame-intervals, one can choose an appropriately long prediction lookahead and tune into new view-streams beforehand to avoid missing the join opportunities.

2.3 Multicasting

Multicasting can drastically reduce the bandwidth required from dedicated media servers. IP multicast, specified decades ago [19], allows sending an IP datagram to a group of hosts identified by a single IP destination address [9]. Hosts may join and leave a multicast group at any time. This requires multicast-capable routers that replicate packets as required. Even though IP multicast is extremely efficient at distributing data to multiple interested receivers, most routers on the Internet keep this functionality turned off due to reasons related to security, billing and the size of the data-structures to be maintained by the router. Nevertheless, the bandwidth conservation benefits of IP multicast have resulted in rising deployment for corporate communications and, more recently, IPTV service.

The seminal work on receiver-driven layered multicast (RLM) [54] focuses on video streaming without interactive pan/tilt/zoom. The authors propose compressing the multimedia signal in hierarchical layers and letting individual receivers choose the layers to join. Receiving more layers leads to better quality. Each layer is delivered using a different multicast group. Note that if a receiver joins too many layers and creates congestion on a link then packets can be dropped indiscriminately from all layers affecting received quality, possibly for multiple receivers that share the congested link. A receiver performs regular tests to decide if it should unsubscribe already joined layers or subscribe new layers. “Shared learning” among receivers can reduce the number of tests and hence the convergence time.

Recently, the RLM framework was adapted for interactive dynamic light field streaming [37]. Depending on the chosen viewpoint, the client decides which views and consequently which multicast groups to subscribe. The latency for joining a new multicast group is generally low with IP multicast [22]. As in the case of RLM, it is the client's responsibility to avoid congestion on intermediate links. The source does not adapt transmission to curtail congestion; it keeps transmitting IP datagrams to the multicast groups' addresses.

Contrary to network-layer IP multicast, P2P streaming implements the multicasting logic in software at the end-hosts rather than routers inside the network [18]. Unlike IP multicast, the application-layer software can be widely deployed with little investment. Although the P2P approach generally results in more duplication of packets and inefficient routing compared to IP multicast, the benefits outweigh the inefficiencies. The source as well as each peer can respond to local retransmission requests as well as perform sophisticated packet scheduling to maximize the experience of downstream peers [61].

P2P streaming systems can be broadly classified into mesh-pull vs. tree-push systems [43]. The design of mesh-pull systems evolved from P2P file-sharing systems. In these systems, a peer advertises the chunks of data that it has and complies with requests to relay chunks to other peers. Tree-push systems, on the other hand, distribute data using one or more complementary trees. After finding its place inside a distribution tree, a peer generally persists to keep its association with the parent and its children and relays data without waiting for requests from children. Generally, tree-push systems result in fewer duplicate packets, lower end-to-end delay and less delay-jitter [8, 46]. These traits are beneficial for interactive streaming systems where select sub-streams of the coded content are required on-the-fly. A tree-based P2P protocol has been recently proposed for interactive streaming of dynamic light fields [39, 40]. Early results demonstrate the capability of the system to support many more users with the same server resources as compared to traditional unicast client-server streaming [40].

3 Spatial-Random-Access-Enabled Video Coding

We have proposed a spatial-random-access-enabled video coding scheme, shown in Fig. 2, in our earlier work [47]. The coded representation consists of multiple resolution layers. 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. 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. Slices are transmitted from the reso-

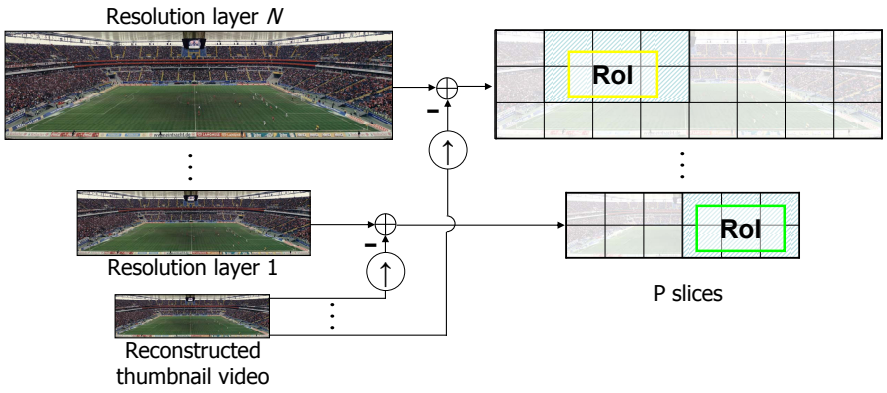

Fig. 2 The thumbnail video constitutes a base layer and is coded with H.264/AVC using I, P and B pictures. The reconstructed base layer video frames are upsampled by a suitable factor and used as prediction signal for encoding video corresponding to the higher resolution layers. Higher resolution layers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

lution layer that 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. Thus, smooth zoom control can be rendered despite storing only few dyadically spaced resolution layers at the server. Note that the encoding takes place once and generates a repository of slices. Relevant slices can be served to several clients depending on their individual RoIs. The encoding can either take place live or offline beforehand.

With the above-mentioned coding scheme, the thumbnail is transmitted continuously. As shown in Fig. 1, the video player can display it to aid navigation. Moreover, the thumbnail can be used for error concealment, in case parts of the RoI do not arrive in time. Ideally, the video delivery system should react to the client's changing RoI with as little latency as possible. The described 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 GOP or having to transmit extra slices from previous frames.

Compliance with State-of-the-Art Video Compression Standards Current video compression standards provide tools like slices but no straightforward method for spatial random access since their main focus has been compression efficiency 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 slices belonging to a high-resolution layer hamper spatial random access. Note that the motion vectors (MVs) can be chosen such that they do not point outside slice boundaries. Also

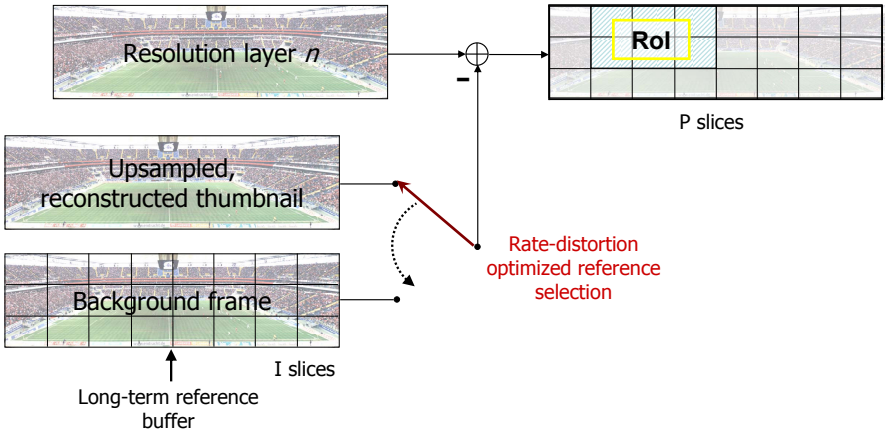

Fig. 3 Each high-resolution layer frame has two references to choose from, the frame obtained by upsampling the reconstructed thumbnail frame and the background frame from the same layer in the background pyramid.

note that instead of SVC, AVC can be employed separately for the high-resolution layers and the MVs can be similarly restricted to eliminate inter-slice dependencies. However, this is very similar to treating the slices as separate video sequences. An obvious drawback is the redundancy between the high-resolution slices and the base layer. A second drawback is that after RoI change, a newly joined slice can only be decoded starting from an intracoded frame. However, if the video player stops displaying the thumbnail video, the transmission of the base layer can be discontinued.

Coding Slices with Multiple Representations Prior work on view random access, mentioned in Sect. 2.1 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 transmit the appropriate representation for a slice depending on the slices that have been transmitted earlier. For some representations, the MVs can be allowed to point outside slice boundaries. Note that this might lower the transmission bit-rate but 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.

Improvement Based on Background Extraction Now let us see how the coding scheme from Fig. 2 can be improved for higher coding efficiency without employing multiple representations. Although the coding scheme of Fig. 2 enables efficient random access, upward prediction using the reconstructed thumbnail frames might result in substantial residual energy for high spatial frequencies. We propose creating a background frame [45, 23] for each high-resolution layer and employing long-term memory motion-compensated prediction (LTM MCP) [75] to exploit the

correlation between this frame and each high-resolution frame to be encoded [48]. The background frame is intracoded. As shown in Fig. 3, 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, relevant I slices from the background frame are transmitted only if they have not been transmitted earlier. This is different from prior work [15] employing a background pyramid, in which the encoder uses only those parts of the background for prediction that exist in the decoder's multi-resolution background pyramid. In [15], the encoder mimics the decoder which builds a background pyramid out of all previously received frames. Note that the camera is likely to be static in such applications since a moving camera might hamper the interactive browsing experience. Background extraction is generally easier with a static camera. Background extraction algorithms as well as detection and update of changed background portions have been previously studied, for example in [30]. Note that the improved coding scheme entails transmitting some I slices from the background frame that might be required for decoding the current high-resolution P slice. Nevertheless, the cost of doing this is amortized over the streaming session. Bit-rate reduction of 70–80% can be obtained with this improvement while retaining efficient random access.

Optimal Slice Size Generally, whenever tiles or slices are employed, choosing the tile size or slice size poses the following trade-off. On one hand, a smaller slice size reduces the overhead of transmitted pixels. The overhead is constituted by pixels that have to be transmitted due to the coarse slice grid but are not used for rendering the display. On the other hand, reducing the slice size worsens the coding efficiency. This is due to increased number of headers and inability to exploit correlation across the slices. The optimal slice size depends on the RoI display dimensions, the dimensions of the high-spatial-resolution video, the content itself and the distribution of the user-selected zoom-factor. Nevertheless, we have demonstrated in prior work that stochastic analysis can estimate the expected number of transmitted pixels per frame [47]. This quantity, denoted by $\psi(s_w, s_h)$, is a function of the slice width, s_w and the slice height, s_h. The average number of bits per pixel required to encode the high-resolution video frame, denoted by $\eta(s_w, s_h)$, can also be observed or estimated as a function of the slice size. The optimal slice size is the one that minimizes the expected number of bits transmitted per frame,

$$(s_w^{\text{opt}}, s_h^{\text{opt}}) = \arg\min_{(s_w, s_h)} \eta(s_w, s_h) \times \psi(s_w, s_h). \tag{1}$$

The results in our earlier work show that the optimal slice size can be determined accurately without capturing user-interaction trajectories [47]. Although the model predicts the optimal slice size accurately, it can underestimate or overestimate the transmitted bit-rate. This is because the popular slices that constitute the salient objects in the video might entail high or low bit-rate 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. Note that the cost function in (1) can be replaced with a Lagrangian cost function that minimizes the weighted sum of the

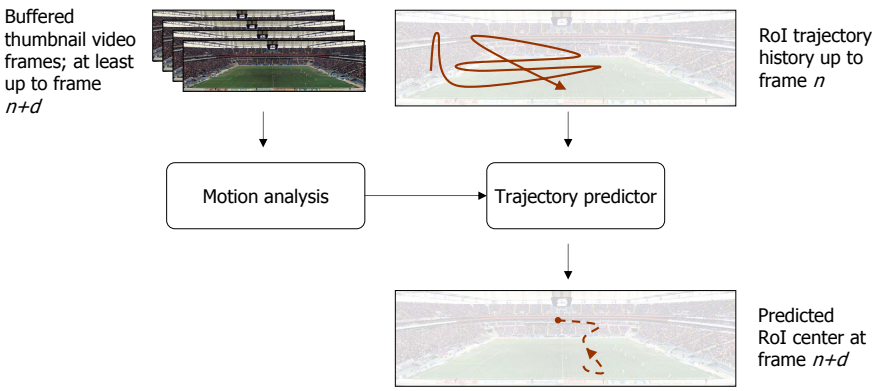

Fig. 4 Video-content-aware RoI prediction analyzes motion in the buffered thumbnail video frames. The video transmission system ensures that some thumbnail video frames are sent ahead of time. Although not shown in the figure, RoI prediction can alternatively be performed at the server. In this case, the server can analyze motion in the high-resolution frames, however, the available trajectory history might be older than current due to network delays. Also, the load on the server increases with the number of clients.

average transmission bit-rate and the incurred storage cost. The storage cost can be represented by an appropriate constant multiplying $\eta(s_w, s_h)$.

4 Pre-Fetching Based on RoI Prediction

The rationale behind pre-fetching is lowering the latency of interaction. Imagine that frame number n is being rendered on the screen. At this point, the user's RoI selection up to frame n has been observed. The goal is to predict the user's RoI at frame $n+d$ ahead of time and pre-fetch relevant slices.

Extrapolating the Navigation Trajectory In our own work [53, 49], we have used an autoregressive moving average (ARMA) model to estimate the velocity of the RoI center:

$$v_t = \alpha v_{t-1} + (1-\alpha)(p_t - p_{t-1}), \tag{2}$$

where, the co-ordinates of the RoI center, observed up to frame n, are given by $p_t = (x_t, y_t)$ for $t = 0, 1 \ldots, n$. The predicted RoI center co-ordinates $\hat{p}_{n+d} = (\hat{x}_{n+d}, \hat{y}_{n+d})$ for frame $n+d$ are given by

$$\hat{p}_{n+d} = p_n + dv_n, \tag{3}$$

suitably adjusted if the RoI happens to veer off the extent of the video frame. The prediction lookahead, d frames, should be chosen by taking into account network

delays and the desired interaction latency. The parameter α above trades off responsiveness to the user's RoI trajectory and smoothness of the predicted trajectory.

Video-Content-Aware RoI Prediction Note that the approach described above is agnostic of the video content. We have explored video-content-aware RoI prediction that analyzes the motion of objects in the video to improve the RoI prediction [53, 49]. The transmission system in this work employs the multi-resolution video coding scheme presented in Sect. 3. The transmission system ensures that some future thumbnail video frames are buffered at the client's side. Figure 4 illustrates client-side video-content-aware RoI prediction. Following are some approaches explored in [53]:

1. Optical flow estimation techniques, for example the Kanade-Lucas-Tomasi (KLT) feature tracker [73], can find feature points in buffered thumbnail frames and track the features in successive frames. The feature closest to the RoI center in frame n can be followed up to frame $n+d$. The location of the tracked feature point can be made the center of the predicted RoI in frame $n+d$ or the predicted RoI can be chosen such that the tracked feature point appears in the same relative location. Alternatively, a smoother trajectory can be obtained by making a change to the RoI center only if the feature point moves more than a certain distance away from the RoI center.
2. Depending on the chosen optical flow estimation technique, the above approach can be computationally intensive. An alternative approach exploits MVs contained in the buffered thumbnail bit-stream. The MVs are used to find a plausible propagation of the RoI center pixel in every subsequent frame up to frame $n+d$. The location of the propagated pixel in frame $n+d$ is deemed to be the center of the predicted RoI. Although the MVs are rate-distortion optimized and might not reflect true motion, the results are competitive to those obtained with the KLT feature tracker [53]. The work in [69] is related in spirit, although the context, mobile augmented reality, is different. In this work, MVs are used to track multiple feature points from one frame to the next while employing homography testing to eliminate outliers among tracked feature points. The algorithm also considers the case of B frames.
3. One can employ multiple RoI predictors and combine their results, for example, through a median operation. This choice guarantees that for any frame interval, if one of the predictors performs poorly compared to the rest, then the median operation does not choose that predictor. In general, the more diversity among the predictors the better.

Compared to the video-content-agnostic schemes, the gain obtained through video-content-aware RoI prediction is higher for longer prediction lookahead d [53]. Moreover, unlike the above approaches that are generic, the motion analysis can be domain-specific [49]. For example, for interactive viewing of soccer, certain objects-of-interest like the ball, the players, the referees, etc. can be tracked and their positions can drive the RoI prediction.

In the approaches above, the user actively controls the input device and the goal of the system is to predict the future path as accurately as possible. In another mode

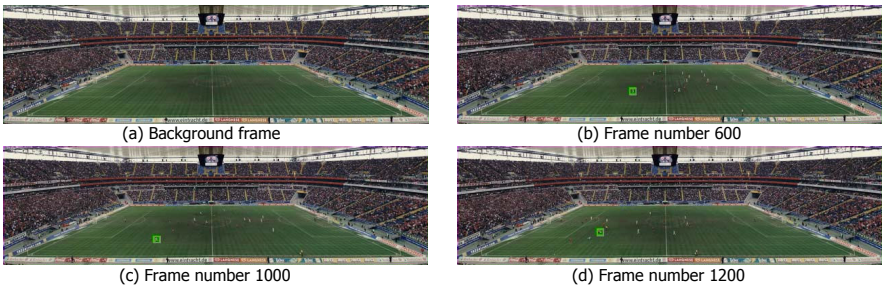

Fig. 5 (a) Background frame. (b)–(d) Example of player tracking. Tracked player is highlighted for better visibility. The frames belong to the decoded thumbnail video having resolution 640×176 pixels. Note that a player is typically less than 5 pixels wide in the thumbnail video.

of operation, the system offers to track a certain object-of-interest for the user such that it relieves navigation burden. In this case, a user-selected trajectory might not be available for comparison or as trajectory history input. In this mode, the goal of the algorithm is to provide a smooth trajectory without deviating from the object.

Figure 5 reproduces a result from [49] that shows the tracking of a soccer player over successive frames of the thumbnail video. The algorithm is based on background subtraction and blob tracking using MVs. Note that a player is typically less than 5 pixels wide in the thumbnail video. Alternatively, the server can process the high-resolution video or the tracking information can be generated through human assistance and trajectories of certain objects-of-interest can be conveyed to the clients to aid their pre-fetching modules.

5 P2P Multicasting for Interactive Region-of-Interest

From the perspective of allowing the system to scale to large numbers of users, it is important to limit both the encoding load as well as the bandwidth required at the server. The video compression approach presented in Sect. 3 limits the encoding load on the server irrespective of the number of users. The goal of this section is to limit the bandwidth required from dedicated servers. We assume that several users are concurrently watching the video, however, each user enjoys independent control of the region to watch. In this section, we review our IRoI P2P streaming system, first introduced in [51, 50], that can achieve P2P live multicast of IRoI video.

5.1 System Architecture

We employ the compression scheme from Sect. 3, illustrated in Figs. 2 and 3, for compressing the thumbnail video and the high-resolution layers. IRoI P2P aims to

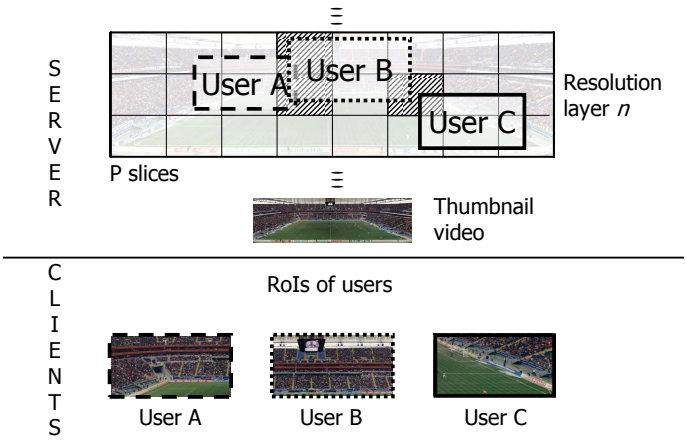

Fig. 6 Example illustrating RoIs of three users within the multi-resolution video representation. The slices shown shaded are commonly wanted by more than one user and represent the "overlaps" exploited by the IRoI P2P system.

exploit overlaps among the users' RoIs. Figure 6 shows overlaps among RoIs of three users. The P2P protocol builds on top of the Stanford Peer-to-Peer Multicast (SPPM) protocol [62, 61] which operates in tree-push manner. SPPM was originally developed for P2P video streaming without any pan/tilt/zoom functionality. Nevertheless, we can leverage SPPM for building and maintaining distribution trees in a distributed manner. Each high-resolution slice, also called enhancement layer slice, is delivered using a separate set of multicast trees. Similarly, multiple complementary multicast trees deliver the thumbnail video, called the base layer. Each peer subscribes the base layer at all times and additionally some enhancement layer slices that are required to render the RoI. Peers also dynamically unsubscribe slices that are no longer required to render the RoI. The RoI prediction lookahead accounts for the latency in joining new trees as well as the playout delay that is employed to mitigate delay jitter among the high-resolution slice packets.

5.2 P2P Protocol

The server maintains a database of slices that each peer is currently subscribed to. Whenever the RoI prediction indicates a change of RoI, the peer sends an RoI-switch request to the server. This consists of the top-left and bottom-right slice IDs of the old RoI as well as the new RoI. In response to the RoI-switch request, the server sends a list of potential parents for every new multicast tree that the peer needs to subscribe. Corresponding to every multicast tree, there is a limit on the number of peers the server can directly serve, and the server includes itself in the list if this quota is not yet full. The server also updates its database assuming that the peer will

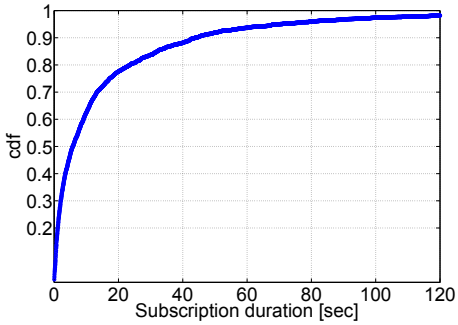

Fig. 7 Cumulative distribution function (cdf) of slice subscription durations for the *Soccer* sequence. The cdf is computed from 1000-second-long user-interaction trajectories of 100 peers. Peer lifetimes themselves are exponentially distributed with an average of 90 seconds. The average slice subscription duration for this sequence is about 16.5 seconds.

be successful in updating its subscriptions. After receiving the list from the server, the peer probes potential parents for every new multicast tree it needs to join. If it receives a positive reply, it sends an attach request for that tree. If it still fails to connect, the peer checks for positive replies from other probed peers and tries attaching to one of them. Once connected to any multicast tree corresponding to a slice, the peer checks if it has previously received the corresponding background I slice. If it has not then the peer obtains the background I slice from one of the peers in the list or the server.

When the RoI prediction indicates a change of RoI, the peer waits a while before sending leave messages to its parents on trees that its RoI no longer requires. This ensures that slices are not unsubscribed prematurely. On the other hand, the peer sends leave messages to its children immediately but keeps forwarding data as long as it receives data from its parent. Upon receiving leave messages, the respective children request potential parents' lists from the server for the respective multicast trees and try finding new parents. The delay in unsubscribing is chosen such that the children experience a smooth handoff from old parent to new parent. In rare cases, a child peer takes longer than the handoff deadline to find a new parent and experiences disruption on that tree. The cumulative distribution function (cdf) of slice subscription durations shown in Fig. 7 indicates how long peers attach to a multicast tree. For the shown example, there are two high-resolution layers apart from the thumbnail video. The RoI is 480×240 pixels whereas the highest resolution layer is 2560×704 pixels. The total number of slices, counting the thumbnail video as one slice and counting slices of the two high-resolution layers, is 382. Each peer subscribes about 24 slices on average corresponding to about 1.1 Mbps bit-rate, whereas the collective bit-rate of all the slices is about 14.1 Mbps.

In addition to leaving multicast trees gracefully, peers can also switch off altogether leading to ungraceful departures. If a child peer does not receive data for a particular tree for a timeout interval, it assumes that the parent is unavailable and tries to rejoin the tree by enquiring about other potential parents. To monitor the online status of parents, peers send Hello messages regularly to their parents and the parents reply back. Since most tree disconnections are graceful and occur due to RoI change, the interval for sending Hello messages can be large to limit the protocol

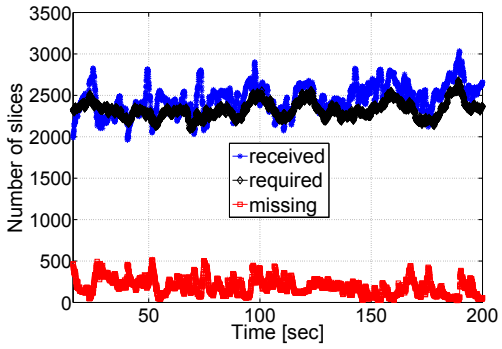

Fig. 8 Trace of received, required and missing slices shown collectively for 100 peers watching the *Soccer* sequence. The percentage of missing slices is about 8.3%. The server was limited to directly serve up to 3 peers per multicast tree. One multicast tree was built per slice. Note that due to the unsubscription delay, peers can receive more slices than required.

overhead. Similar to SPPM, a loop-avoidance mechanism on individual distribution trees ensures that a descendant is not chosen as a parent [62, 64, 63, 11, 61]. For additional details on peer state transitions and timeouts associated with sending and receiving control messages, the reader may refer to [60].

The server advances the base layer transmission slightly compared to the transmission of the enhancement layer slices. This way peers can buffer some base layer frames as well as request retransmissions of lost base layer packets. The stringent latency constraint associated with interactive RoI makes retransmissions of enhancement layer packets difficult. Recall that the base layer can be used to fill in missing parts while rendering the RoI. The error-concealed parts might appear blurry but the user experiences low-latency RoI control.

5.3 Protocol Performance

A simulation with 100 peers was carried out by implementing the IRoI P2P protocol within the NS-2 network simulator. The shape of the cdf of peer uplink capacities was modeled after the one presented in [55], however, the average of the peer uplink capacities was set to 2 Mbps, slightly higher than the 1.7 Mbps average reported in [55]. A single tree was built per slice. The average upper bound of PSNR among the peers was 41.9 dB. This corresponds to the hypothetical case when each peer receives all the slices that it needs. The average lower bound of PSNR among the peers was 30.6 dB assuming that the base layer is successfully received. The lower bound corresponds to the case when no high-resolution slices are received by the peers and the RoI is rendered only using the base layer. The average PSNR was found to be 38.6 dB, indicating that peers receive most of the enhancement layer slices required to render respective RoIs.

Figure 8 shows the trace of received, required, and missing slices collectively for the 100 peers. The percentage of missing slices is about 8.3%. The server was limited to directly serve up to 3 peers per multicast tree. Note that without such a limit, the server's capacity might be exhausted and the system might not be able to

supply a new slice that no peer currently subscribes. Interestingly, the average number of slices with non-zero fan-out is only about 172 indicating that all slices are not streamed all the time. The load on the server was about 13.7 Mbps which is less than the 14.1 Mbps bit-rate of the multi-resolution representation. Another simulation was carried out in which two multicast trees were built per slice delivering odd and even frames respectively. The percentage of missing slices remained roughly the same, however, it was observed that for about 65% missing slices, the corresponding slice from the previous frame was available. This allows error-concealment using pixels from the previous frame in most cases, thus maintaining high spatial resolution, which is important for virtual pan/tilt/zoom. The picture quality was better even though the average PSNR improved by only about 0.05 dB. In experiments with other high-spatial-resolution video sequences, the average PSNR improved by about 1–2 dB compared to single tree per slice. The protocol overhead due to control messages was observed to be between 5–10% of the total traffic.

5.4 Server Bandwidth Allocation

The slices hosted by the server constitute a set of P2P multicast streams which generally vary in popularity. A framework for server bandwidth allocation among multiple P2P multicast streams has been proposed in a related thread of our research [50, 52]. The framework accommodates multiple multicast trees per stream and can take into account the popularity, the rate-distortion operating point as well as the peer churn rate associated with each stream. The framework allows minimizing different metrics like mean distortion among the peers, number of frame-freezes overall, etc. When the available server bandwidth is scarce, it is very important to judiciously allocate rate to the most important slices. For the above example with 100 peers and 2 trees per slice, the server capacity was set to 10 Mbps and the limits on the numbers of direct children associated with the multicast trees were computed by minimizing expected mean distortion. Note that the 10 Mbps server capacity is less than the 14.1 Mbps bit-rate of the multi-resolution representation. The optimized rate allocation among the slices was compared against a heuristic scheme that sequentially allocates rate to slices with ascending slice IDs, stopping when the capacity exhausts. The optimized rate allocation resulted in about 21% missing slices whereas the heuristic scheme resulted in about 82% missing slices.

6 Conclusions

In this chapter, we have reviewed the technical challenges that must be overcome to watch high-resolution video with interactive pan/tilt/zoom and possible solutions. IRoI video streaming allows watching user-selected portions of a high-resolution video even on displays of lower spatial resolution.

In the remote streaming scenario, the transmission of the entire high-resolution video is generally not possible due to bandwidth limitations. Broadly speaking, there are two approaches to provide a video sequence as controlled by the user's pan/tilt/zoom commands. The RoI video sequence can either be cropped from the raw high-resolution video and encoded prior to transmission or the adopted compression format can allow easy extraction of the relevant portions from the compressed representation. The first approach possesses the drawback that RoI video encoding has to be performed for each user separately. Additionally, if the high-spatial-resolution video is not available in the raw format and the users watch the sequence asynchronously, the high-spatial-resolution video has to be decoded before cropping the RoI sequence for each user.

Spatial-random-access-enabled video compression limits the load of encoding to a one-time encoding of the video, possibly with multiple resolution layers to support continuous zoom. This is beneficial for streaming both live content as well as pre-stored content to multiple users. Even when the video is played back from locally stored media, a different RoI trajectory has to be accommodated each time a user watches the content. Spatial random access allows the video player to selectively decode relevant regions only. This chapter presents a spatial-random-access-enabled video coding scheme in detail that allows the receiver to start decoding a new region, with an arbitrary zoom factor, during any frame interval instead of having to wait for the end of a GOP or having to transmit extra slices from previous frames. Background extraction can be used with such a coding scheme to reduce transmission bit-rate as well as the size of the stored video. We also show how to choose the slice size to attain the right balance between storage and mean transmission bit-rate.

Pre-fetching helps to reduce the latency of interaction. Irrespective of whether pre-fetching is employed or not, having a base layer helps render missing parts of the RoI. This way, the system can always render the RoI chosen by the user, thus offering accurate and low-latency RoI control. The chapter presents several RoI prediction techniques for pre-fetching. Some techniques are employed at the client, some at the server and some are distributed between the server and the client. For example, the server can send the trajectories of key objects in the video to the clients to aid their RoI prediction modules.

This chapter also shows how to use P2P streaming to drastically reduce the bandwidth required from the server for supporting increasing numbers of users. It is crucial for this approach that the P2P overlay reacts quickly to the changing RoIs of the peers and limits the disruption due to the changing relationships among the peers. The IRoI P2P protocol presented in this chapter makes sure that a child-peer experiences smooth transition from old parent to new parent when the old parent willfully unsubscribes a multicast tree that is no longer required for its RoI. Typically, when users choose regions from a high-spatial-resolution video, some regions are more popular than others. It is very important, especially when the server has limited bandwidth, to judiciously allocate the available rate among the regions streamed by the server.

Spatial-random-access-enabled video coding plays an important role in the P2P distribution system. It simplifies the peer's task of choosing which multicast trees

to join on-the-fly. A scheme based on multiple representations coding of the slices might further reduce the download rate required by each peer. However, such a scheme might reduce the degree of overlaps and affect the gains possible from the P2P approach apart from requiring more storage space at the server. The IRoI P2P system presented in this chapter assumes that peers watch the video synchronously. If peers can watch any time-segment, i.e., rewind and fast-forward then the relevant data could still be retrieved from each others' cache. Since storage is becoming cheaper, the cache size employed by the peers for storing previously received content can be assumed to be reasonably large. Such kind of "time-shifted streaming" can be looked upon as the temporal counterpart of the spatial freedom provided by pan/tilt/zoom. A system providing both functionalities would be a natural extension of the system presented here.

Acknowledgements Fraunhofer Heinrich-Hertz Institute (HHI), Berlin, Germany generously provided the *Soccer* sequence.

References

1. Dodeca 2360: An omni-directional video camera providing over 100 million pixels per second by Immersive Media. URL `http://www.immersivemedia.com`. Seen on Sept. 16, 2009
2. Halo: Video conferencing product by Hewlett-Packard. URL `http://www.hp.com/halo/index.html`. Seen on Sept. 16, 2009
3. Video clip showcasing interactive TV with pan/tilt/zoom. URL `http://www.youtube.com/watch?v=Ko9jcIjBXnk`. Seen on Sept. 26, 2009
4. H.264/AVC/MPEG-4 Part 10 (ISO/IEC 14496-10: Advanced Video Coding). Standard (2003)
5. ISO/IEC 15444-1:2004, JPEG2000 Specification. Standard (2004)
6. Annex G of H.264/AVC/MPEG-4 Part 10: Scalable Video Coding (SVC). Standard (2007)
7. Aaron, A., Ramanathan, P., Girod, B.: Wyner-Ziv coding of light fields for random access. In: Proc. IEEE 6th Workshop on Multimedia Signal Processing (MMSP'04), pp. 323–326. Siena, Italy (2004)
8. Agarwal, S., Singh, J., Mavlankar, A., Baccichet, P., Girod, B.: Performance of P2P live video streaming systems on a controlled test-bed. In: Proc. of 4th Intl. Conf. on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM'08). Innsbruck, Austria (2008)
9. Albanna, Z., Almeroth, K., Meyer, D., Schipper, M.: IANA guidelines for IPv4 multicast address assignments. RFC 3171 (2001)
10. Azuma, R., Bishop, G.: A frequency-domain analysis of head-motion prediction. In: Proc. ACM SIGGRAPH'95, pp. 401–408. Los Angeles, CA, USA (1995)
11. Baccichet, P., Noh, J., Setton, E., Girod, B.: Content-aware P2P video streaming with low latency. In: Proc. IEEE Intl. Conf. on Multimedia and Expo (ICME'07), pp. 400–403. Beijing, China (2007)
12. Baccichet, P., Zhu, X., Girod, B.: Network-aware H.264/AVC region-of-interest coding for a multi-camera wireless surveillance network. In: Proc. Picture Coding Symposium (PCS'06). Beijing, China (2006)
13. Bauermann, I., Steinbach, E.: RDTC optimized compression of image-based scene representations (Part I): Modeling and theoretical analysis. IEEE Trans. Image Process. **17**(5), 709–723 (2008)

14. Bauermann, I., Steinbach, E.: RDTC optimized compression of image-based scene representations (Part II): Practical coding. IEEE Trans. Image Process. **17**(5), 724–736 (2008)
15. Bernstein, J., Girod, B., Yuan, X.: Hierarchical encoding method and apparatus employing background references for efficiently communicating image sequences (1992)
16. Cheung, G., Ortega, A., Cheung, N.M.: Generation of redundant frame structure for interactive multiview streaming. In: Proc. IEEE 17th Packet Video Workshop. Seattle, WA, USA (2009)
17. Cheung, N.M., Ortega, A., Cheung, G.: Distributed source coding techniques for interactive multiview video streaming. In: Proc. Picture Coding Symposium (PCS'09). Chicago, IL, USA (2009)
18. Chu, Y.H., Rao, S., Seshan, S., Zhang, H.: A case for end system multicast. IEEE J. Sel. Areas Commun. **20**(8), 1456–1471 (2002)
19. Deering, S.: Host extensions for ip multicasting. RFC 1112 (1989)
20. Devaux, F., Meessen, J., Parisot, C., Delaigle, J., Macq, B., Vleeschouwer, C.D.: A flexible video transmission system based on JPEG2000 conditional replenishment with multiple references. In: Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'07), vol. 1, pp. 825–828. Honolulu, HI, USA (2007)
21. Dhondt, Y., Lambert, P., Notebaert, S., Van de Walle, R.: Flexible macroblock ordering as a content adaptation tool in H.264/AVC. In: Proc. SPIE Conf. on Multimedia Systems and Applications VIII, vol. 6015, pp. 601,506.1–601,506.9. Boston, MA, USA (2005)
22. Estrin, D., Handley, M., Helmy, A., Huang, P., Thaler, D.: A dynamic bootstrap mechanism for rendezvous-based multicast routing. In: Proc. IEEE INFOCOM, pp. 1090–1098. New York, USA (1999)
23. Farin, D., de With, P., Effelsberg, W.: Robust background estimation for complex video sequences. In: Proc. IEEE Intl. Conf. on Image Processing (ICIP'03), vol. 1, pp. 145–148 (2003)
24. Fehn, C., Weissig, C., Feldmann, I., Mueller, M., Eisert, P., Kauff, P., Bloss, H.: Creation of high-resolution video panoramas of sport events. In: Proc. 8th IEEE Intl. Symp. on Multimedia (ISM'06), pp. 291–298. San Diego, CA, USA (2006)
25. Flierl, M., Mavlankar, A., Girod, B.: Motion and disparity compensated coding for multi-view video. IEEE Trans. Circuits Syst. Video Technol. **17**(11), 1474–1484 (2007). (Invited Paper)
26. Girod, B.: The efficiency of motion-compensating prediction for hybrid coding of video sequences. IEEE J. Sel. Areas Commun. **5**(7), 1140–1154 (1987)
27. Girod, B.: Motion-compensating prediction with fractional-pel accuracy. IEEE Trans. Commun. **41**(4), 604–612 (1993)
28. Girod, B.: Efficiency analysis of multihypothesis motion-compensated prediction for video coding. IEEE Trans. Image Process. **9**(2), 173–183 (2000)
29. Gruenheit, C., Smolic, A., Wiegand, T.: Efficient representation and interactive streaming of high-resolution panoramic views. In: Proc. IEEE Intl. Conf. on Image Processing (ICIP'02), vol. 3, pp. III–209–III–212. Rochester, NY, USA (2002)
30. Hepper, D.: Efficiency analysis and application of uncovered background prediction in a low bit rate image coder. IEEE Trans. Commun. **38**(9), 1578–1584 (1990)
31. Heymann, S., Smolic, A., Mueller, K., Guo, Y., Rurainsky, J., Eisert, P., Wiegand, T.: Representation, coding and interactive rendering of high-resolution panoramic images and video using MPEG-4. In: Proc. Panoramic Photogrammetry Workshop (PPW'05). Berlin, Germany (2005)
32. Jagmohan, A., Sehgal, A., Ahuja, N.: Compression of lightfield rendered images using coset codes. In: Proc. 37th Annual Asilomar Conference on Signals, Systems, and Computers, vol. 1 (2003)
33. Kauff, P., Schreer, O.: Virtual team user environments - a step from tele-cubicles towards distributed tele-collaboration in mediated workspaces. In: Proc. IEEE Intl. Conf. on Multimedia and Expo (ICME'02), vol. 2, pp. 9–12. Lausanne, Switzerland (2002)
34. Kimata, H., Kitahara, M., Kamikura, K., Yashima, Y., Fujii, T., Tanimoto, M.: Low-delay multiview video coding for free-viewpoint video communication. Syst. Comput. Japan **38**(5), 14–29 (2007)
35. Kiruluta, A., Eizenman, M., Pasupathy, S.: Predictive head movement tracking using a kalman filter. IEEE Trans. Syst., Man and Cybernetics, Part B: Cybernetics **27**(2), 326–331 (1997)

36. Kopf, J., Uyttendaele, M., Deussen, O., Cohen, M.F.: Capturing and viewing gigapixel images. In: Proc. ACM SIGGRAPH'07, vol. 26, pp. 93.1–93.10. San Diego, CA, USA (2007)
37. Kurutepe, E., Civanlar, M.R., Tekalp, A.M.: A receiver-driven multicasting framework for 3DTV transmission. In: Proc. 13th European Signal Processing Conference (EUSIPCO'05). Antalya, Turkey (2005)
38. Kurutepe, E., Civanlar, M.R., Tekalp, A.M.: Interactive transport of multi-view videos for 3DTV applications. Journal of Zhejiang University - Science A **7**(5), 830–836 (2006)
39. Kurutepe, E., Sikora, T.: Feasibility of multi-view video streaming over p2p networks. In: 3DTV Conf.: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON'08), pp. 157–160 (2008)
40. Kurutepe, E., Sikora, T.: Multi-view video streaming over p2p networks with low start-up delay. In: Proc. IEEE Intl. Conf. on Image Processing (ICIP'08), pp. 3088–3091 (2008)
41. Levoy, M., Hanrahan, P.: Light field rendering. In: Proc. ACM SIGGRAPH'96, pp. 31–42. New Orleans, LA, USA (1996)
42. Liu, Y., Huang, Q., Zhao, D., Gao, W.: Low-delay view random access for multi-view video coding. In: Proc. IEEE Intl. Symp. on Circuits and Systems (ISCAS'07), pp. 997–1000. New Orleans, LA, USA (2007)
43. Magharei, N., Rejaie, R., Guo, Y.: Mesh or Multiple-Tree: A Comparative Study of Live Peer-to-Peer Streaming Approaches. In Proc. IEEE INFOCOM (2007)
44. Makar, M., Mavlankar, A., Girod, B.: Compression-aware digital pan/tilt/zoom. In: Proc. 43rd Annual Asilomar Conference on Signals, Systems, and Computers (2009)
45. Massey, M., Bender, W.: Salient stills: Process and practice. IBM Systems Journal **35**(3&4), 557–573 (1996)
46. Mavlankar, A., Baccichet, P., Girod, B., Agarwal, S., Singh, J.: Video quality assessment and comparative evaluation of peer-to-peer video streaming systems. In: Proc. IEEE Intl. Conf. on Multimedia and Expo (ICME'08), pp. 645–648. Hanover, Germany (2008)
47. Mavlankar, A., Baccichet, P., Varodayan, D., Girod, B.: Optimal slice size for streaming regions of high resolution video with virtual pan/tilt/zoom functionality. In: Proc. 15th European Signal Processing Conference (EUSIPCO'07), pp. 1275–1279. Poznan, Poland (2007)
48. Mavlankar, A., Girod, B.: Background extraction and long-term memory motion-compensated prediction for spatial-random-access-enabled video coding. In: Proc. Picture Coding Symposium (PCS'09). Chicago, IL, USA (2009)
49. Mavlankar, A., Girod, B.: Pre-fetching based on video analysis for interactive region-of-interest streaming of soccer sequences. In: Proc. IEEE Intl. Conf. on Image Processing (ICIP'09). Cairo, Egypt (2009). (Accepted)
50. Mavlankar, A., Noh, J., Baccichet, P., Girod, B.: Optimal server bandwidth allocation for streaming multiple streams via P2P multicast. In: Proc. IEEE 10th Workshop on Multimedia Signal Processing (MMSP'08). Cairns, Australia (2008)
51. Mavlankar, A., Noh, J., Baccichet, P., Girod, B.: Peer-to-peer multicast live video streaming with interactive virtual pan/tilt/zoom functionality. In: Proc. IEEE Intl. Conf. on Image Processing (ICIP'08), pp. 2296–2299. San Diego, CA, USA (2008)
52. Mavlankar, A., Noh, J., Baccichet, P., Girod, B.: Optimal server bandwidth allocation among multiple P2P multicast live video streaming sessions. In: Proc. IEEE 17th Packet Video Workshop. Seattle, WA, USA (2009)
53. Mavlankar, A., Varodayan, D., Girod, B.: Region-of-interest prediction for interactively streaming regions of high resolution video. In: Proc. IEEE 16th Packet Video Workshop, pp. 68–77. Lausanne, Switzerland (2007)
54. McCanne, S., Jacobson, V., Vetterli, M.: Receiver-driven layered multicast. In: Proc. ACM SIGCOMM. Stanford, CA, USA (1996)
55. Noh, J., Baccichet, P., Girod, B.: Experiences with a large-scale deployment of stanford peer-to-peer multicast. In: Proc. IEEE 17th Packet Video Workshop. Seattle, WA, USA (2009)
56. Ramanathan, P., Girod, B.: Random access for compressed light fields using multiple representations. In: Proc. IEEE 6th Intl. Workshop on Multimedia Signal Processing (MMSP'04), pp. 383–386. Siena, Italy (2004)

57. Ramanathan, P., Girod, B.: Rate-distortion optimized streaming of compressed light fields with multiple representations. In: Proc. 14th Packet Video Workshop. Irvine, CA, USA (2004)
58. Ramanathan, P., Kalman, M., Girod, B.: Rate-distortion optimized interactive light field streaming. IEEE Trans. Multimedia **9**(4), 813–825 (2007)
59. Schwarz, H., Marpe, D., Wiegand, T.: Overview of the scalable video coding extension of the H.264/AVC standard. IEEE Trans. Circuits Syst. Video Technol. **17**(9), 1103–1120 (2007)
60. Setton, E.: Congestion-aware video streaming over peer-to-peer networks. Ph.D. thesis, Stanford University, Stanford, CA, USA (2006)
61. Setton, E., Baccichet, P., Girod, B.: Peer-to-peer live multicast: A video perspective. Proc. IEEE **96**(1), 25–38 (2008)
62. Setton, E., Noh, J., Girod, B.: Rate-distortion optimized video peer-to-peer multicast streaming. In: Proc. Workshop on Advances in Peer-to-Peer Multimedia Streaming at ACM Multimedia, pp. 39–48. Singapore (2005). Invited paper
63. Setton, E., Noh, J., Girod, B.: Congestion-distortion optimized peer-to-peer video streaming. In: Proc. IEEE Intl. Conf. on Image Processing (ICIP'06), pp. 721–724. Atlanta, GA, USA (2006)
64. Setton, E., Noh, J., Girod, B.: Low latency video streaming over peer-to-peer networks. In: Proc. IEEE Intl. Conf. on Multimedia and Expo (ICME'06), pp. 569–572. Toronto, Canada (2006)
65. Shum, H.Y., Kang, S.B., Chan, S.C.: Survey of image-based representations and compression techniques. IEEE Trans. Circuits Syst. Video Technol. **13**(11), 1020–1037 (2003)
66. Singhal, S.K., Cheriton, D.R.: Exploiting position history for efficient remote rendering in networked virtual reality. Presence: Teleoperators and Virtual Environments **4**, 169–193 (1995)
67. Smolic, A., McCutchen, D.: 3DAV exploration of video-based rendering technology in MPEG. IEEE Trans. Circuits Syst. Video Technol. **14**(3), 348–356 (2004)
68. Smolic, A., Mueller, K., Merkle, P., Fehn, C., Kauff, P., Eisert, P., Wiegand, T.: 3D Video and Free Viewpoint Video - Technologies, Applications and MPEG Standards. In: Proc. IEEE Intl. Conf. on Multimedia and Expo (ICME'06), pp. 2161–2164. Toronto, ON, Canada (2006)
69. Takacs, G., Chandrasekhar, V., Girod, B., Grzeszczuk, R.: Feature tracking for mobile augmented reality using video coder motion vectors. In: 6th IEEE and ACM Intl. Symp. on Mixed and Augmented Reality (ISMAR'07), pp. 141–144. Nara, Japan (2007)
70. Tanimoto, M.: Free viewpoint television — FTV. In: Proc. Picture Coding Symposium (PCS'04). San Francisco, CA, USA (2004)
71. Taubman, D., Prandolini, R.: Architecture, philosophy and performance of JPIP: Internet protocol standard for JPEG2000. In: Proc. SPIE Intl. Symp. on Visual Communications and Image Processing (VCIP'03), vol. 5150, pp. 649–663. Lugano, Switzerland (2003)
72. Taubman, D., Rosenbaum, R.: Rate-distortion optimized interactive browsing of JPEG2000 images. In: Proc. IEEE Intl. Conf. on Image Processing (ICIP'00), pp. 765–768. Barcelona, Spain (2000)
73. Tomasi, C., Kanade, T.: Detection and tracking of point features. Tech. Rep. CMU-CS-91-132, Carnegie Mellon University, Pittsburgh, PA (1991)
74. Wiegand, T., Sullivan, G., Bjontegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE Trans. Circuits Syst. Video Technol. **13**(7), 560–576 (2003)
75. Wiegand, T., Zhang, X., Girod, B.: Long-term memory motion-compensated prediction. IEEE Trans. Circuits Syst. Video Technol. **9**(1), 70–84 (1999)

Index