Bandwidth-Efficient Interactive Multiview Live Video Streaming using Redundant Frame Structures

Gene Cheung*, Antonio Ortega† and Ngai-Man Cheung†
* Hewlett-Packard Laboratories Japan, 3-29-21, Takaido-higashi, Suginami-ku, 168-8585 Tokyo Japan
E-mail: gene-cs.cheung@hp.com Tel: +81-3-4316-2829
† Signal and Image Processing Institute, University of Southern California, Los Angeles, CA 90089
E-mail: {antonio.ortega, ngai-man.cheung}@sipi.usc.edu Tel: (213) 740-2320

***Abstract*—We address the problem of real-time encoding multiview video for interactive multiview live video streaming, where clients can interactively switch views during video playback. In other words, as a client is playing back successive frames (in time), it sends requests to the server to switch to different views while continuing uninterrupted temporal playback. Noting that standard tools for random access (i.e., I-frame insertion) can be bandwidth-inefficient for view switching, we propose an algorithm to select and encode an optimal combination of I- and/or redundant P-frames in order to facilitate view switching for a given computation constraint. We show in our experiments that we can generate redundant frame structures offering a range of useful tradeoff points between transmission and computation, including ones that outperform simple I-frame insertion structures by up to 32.9% in terms of bandwidth efficiency, if we are willing to double computational resources devoted to encoding.**

I. Introduction

Multiview video consists of sequences of spatially related pictures captured simultaneously and periodically by multiple closely spaced cameras. Much of the previous research on multiview video focuses on compression, where the goal is optimized rate-distortion performance to encode all views across all time jointly [1], [2].

In this paper, we focus instead on the *interactive multiview live video streaming* (IMVS-live) scenario, where a frame structure for live-captured multiview video is chosen and frames are encoded in real-time into the structure by a server, during a live streaming session, for streaming clients to *interactively* request desired views. More specifically, each client watches and requests one view at a time out of many available views, while the video is played forward in time; this results in a different traversal of views across time for each client. The IMVS-live server must be capable of supporting a possibly large group of streaming clients, each with its own unique view traversal, given limited computation resources.

Our objective is to design a frame structure to support a desired level of view interactivity for streaming clients in IMVS-live while minimizing expected transmission bandwidth, given a fixed computation resource. The level of view interactivity is determined by the *view switching period* M; i.e., view switching can only take place at multiples of M frames.

A natural approach to enable this kind of interactive view switching is to make use of standard random access tools, i.e., making every M-th frame (in all views) an I-frame. Our work is based on the observation that *random access and view switching are fundamentally different functionalities*, and thus efficient tools for one problem may not provide the best solution for the other. For random access to a frame, one can make no assumptions about which frames are available at the decoder; independently coded I-frames are therefore well suited for this purpose. View switching, on the other hand, arises when temporal playback is not interrupted, i.e., successive frames are displayed, but one wishes to switch point of view. The key difference is that the decoder has access to some of the frames immediately preceding the requested frame (albeit from a different view) in time. Thus, since consecutive frames in different views tend to be correlated, using an I-frame for switching can be bandwidth-inefficient.

The main focus of our work is then to study alternatives for view switching that are more bandwidth-efficient than simple I-frame insertions. Note that our proposed tools *do not* support random access, which is needed for late clients to join live streaming sessions already in progress. Thus, random access I-frames can be inserted at *join session period* M', in combination with our proposed view-switching tools, and it will be up to the system designer to select the appropriate parameters M and M' for a given live application, where typically $M \ll M'$.

The design of a multiview frame structure to permit view interactivity in IMVS-live involves a tradeoff between the expected transmission rate and the computation cost required to encode frames into the structure. At one extreme, one could encode every switching point as an I-frame (and subsequent $M-1$ frames in the same view as P-frames), so that only K encoders are required for K total views. In this case, clients play back frames produced by one of the encoders and switching views can be done by switching encoders at frame positions that are multiples of M. While computation cost is minimal (K is the minimum number of encoders needed to produce K independently encoded views), the frequent transmission of I-frame can be bandwidth-inefficient. At another extreme, a system could be designed with dedicated encoders for each client, so that as a client interactively selects a view traversal, the corresponding set of frames is encoded. While this results in minimum transmission cost (view switching is achieved via a cross-view P-frame), the computation required is prohibitive when the number of clients is large.

Clearly, between these two extremes more practical multiview frame structures can be selected to obtain different trade offs between transmission and computation costs. Recently, for non-live IMVS scenarios, redundant P-frames [3] and distributed source coding (DSC) frames [4] have been used to trade off transmission with storage costs. In a live scenario, we observe that they offer similar tradeoffs between expected transmission cost and volume of data generated (and hence corresponding computation required), and hence are suitable as view-switching tools for IMVS-live as well. In this paper, we develop an algorithm that finds a good frame structure using redundant P-frames and I-frames trading off transmission and computation. We show in our experiments that our algorithm can offer a range of useful tradeoff points; in particular, we show that in some cases our algorithm generated structures reducing expected transmission rate by up to 32.9% as compared to I-frame insertion, while requiring double encoder computation resources.

The outline of the paper is as follows. We first review related work in Section II. We then discuss IMVS-live system and models in Section III. We formulate the problem of designing a multiview frame structure for IMVS-live in Section IV and present our algorithm in Section V. We discuss our experiments in Section VI.

II. Related Work

As mentioned, much of the previous research in multiview video has focused on efficient compression of all frames of all views [1], [2]. Using such structures directly to facilitate view switching in IMVS-live poses two problems. First, each client watches only one view at a time, thus compressing and sending frames of more than one view (which may be required for correct decoding at client due to cross-view prediction) is bandwidth-inefficient. Second, even if one eliminates cross-view predictions and inserts random access I-frames for all views every M-frame interval, server will still need to send a leading I-frame every M frames for each view switch. Clearly, for small M this again leads to bandwidth inefficiency.

In contrast, we study the bandwidth-optimal design of frame structures for interactive view switching during live streaming. Specifically, our formulation explicitly seeks to minimize the expected transmission rate during an IMVS-live session for given desired view interactivity, at the expense of a modest and controlled increase in computation.

Study of the conflicting requirements of interactivity and compression has been considered in the context of interactive light field streaming using DSC [5], SP-frames [6], and *rerouting* [7], where multiple blocks are transmitted to correctly decode and display a single block. Our IMVS-live work differs in that we allow *multiple decoded versions* of a single original picture (at the expense of increased in computation), so that multiple decoding paths demanding the same view frame can each have their own versions of inter-coded frames, without resorting to a more transmission-expensive intra-coded frame.

We have previously posed the IMVS problem for the offline encoding scenario, where content is pre-encoded *a priori* and streamed to clients at a later time. In that case we considered the tradeoff between transmission and storage costs using redundant P-frames [3] and DSC frames [4], whereas here, for the IMVS-live scenario, tradeoffs between transmission and computation costs are considered.

III. System & Interaction Models

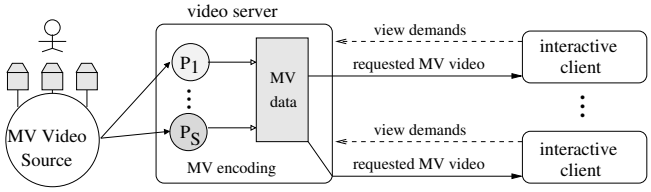


Fig. 1. Interactive Multiview Live Video Streaming System

A. System Model

The system model we consider for the IMVS-live scenario is shown in Fig. 1. A *Multiview Video Source* simultaneously captures multiple pictures of different views at regular intervals. An example of a multiview sequence of two views across four time instants is shown in Fig. 2(a). A *Video Server* sequentially grabs the captured uncompressed pictures from Multiview Video Source and encodes them in real-time based on a frame structure designed *a priori* by an algorithm to be discussed in Section V. Using the same encoded MV frame structure $\mathcal{T}$, Video Server can serve a large number of streaming clients without further encoding/transcoding.

B. View Interaction Model

In what follows, we will use the term *frame* to denote a specific coded version of a picture and use the term *picture* for the corresponding original captured image. Thus, our system will have redundant encodings, in the sense that there may be multiple frames representing a given picture. We assume a view interaction model where, upon watching any decoded version of the picture $F^o_{i,j}$, corresponding to time instant i and view j, an interactive client will request a coded version of picture $F^o_{i+1,k}$ of view k and *next* time instant $i+1$, where $j-1 \leq k \leq j+1$, with *view transition probability* $\alpha_j(k)$; we call this interactivity *forward view switching*[1].

Note that a significant difference between our setting and that of interactive light field streaming [5], [6], [7] is that in the latter case the user is free to explore a static scene in all directions, while here we play forward in time with only limited switching possibilities (i.e., among neighboring views).

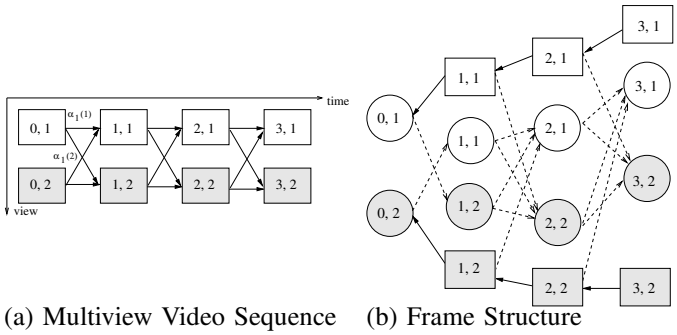
Though our interactive model presumes a client's desire to switch views at single-frame level, our model encompasses the more general case of a view switching period M; in that case a "frame" $F_{i,j}$ in our model would represent M consecutive frames in the same view j (a carefully chosen P- or I-frame determined by our optimization followed by $M-1$ consecutive P-frames of the same view).

[1]We assume here that continuous forward playback of live video is desirable, so that viewers are synchronized in time with the live event.

IV. IMVS-LIVE PROBLEM FORMULATION

We now formulate our IMVS-live problem. We present definitions of key terms, then formally define the optimization.

A. Definitions



(a) Multiview Video Sequence (b) Frame Structure

Fig. 2. Example of Redundant Frame Structure. I- and P-frames are drawn as circles and rectangles, respectively. A solid edge from $F_{i+1,k}$ to $F_{i,j}$ in (b) means a P-frame $F_{i+1,k}$ is predictively coded using reference frame $F_{i,j}$. A dotted edge from $F_{i,j}$ to $F_{i+1,k}$ in (b) means a schedule $\mathbf{G}$ dictates transition to $F_{i+1,k}$ if view k is requested after viewing $F_{i,j}$.

1) Redundant Frame Structure: We assume we are designing a structure *a priori* for a to-be-encoded multiview sequence of K views and N time switching instants[2]. Assume also that view transition probabilities $\alpha_j(k)$'s are known *a priori*, that each starting frame, $F_{0,k}$, has probability w_k, and each $F_{i,j}$ can only transition to frames of neighboring views $F_{i+1,k}$, $\max(1, j-1) \leq k \leq \min(K, j+1)$. Fig. 2(a) shows an example multiview sequence where $N = 3$ and $K = 2$.

One can construct a *redundant frame structure* $\mathcal{T}$ comprised of coded I- and P-frames, denoted as $I_{i,j}$'s and $P_{i,j}$'s, to represent a multiview sequence and enable IMVS-live. Here a P-frame $P_{i,j}$ is a predictively coded frame using a selected $F_{i-1,k}$ as predictor. A structure representing the example multiview sequence in Fig. 2(a) is shown in Fig. 2(b).

A structure $\mathcal{T}$ forms a set of dependency trees with I-frames as root nodes. $\mathcal{T}$ is redundant in that an original picture $F^o_{i,j}$ can be represented by multiple frames $F_{i,j}$'s. In Fig. 2(b), $F^o_{2,1}$ is represented by P-frame $P_{2,1}$ and I-frame $I_{2,1}$. Multiple $F_{i,j}$'s are used to avoid coding drift.

2) Frame-to-frame Schedule: Associated with a structure $\mathcal{T}$ is a *frame-to-frame schedule* $\mathbf{G}$, which determines which frame $F_{i+1,k}$ should be sent by the server, given that the viewer has just observed $F_{i,j}$ and requested view k. We denote a scheduled frame-to-frame transition as $F_{i,j} \overset{\mathbf{G}}{\Rightarrow} F_{i+1,k}$. For given structure $\mathcal{T}$, there exists only one[3] logical schedule $\mathbf{G}$ described below. For desired view k, if there exists a P-frame $P_{i+1,k}$ predicted from $F_{i,j}$, then schedule $\mathbf{G}$ will schedule $P_{i+1,k}$ since it is for this transition that $P_{i+1,k}$ was constructed. If $P_{i+1,k}$ predicted from $F_{i,j}$ does not exist, then $\mathbf{G}$ would schedule an I-frame $I_{i+1,k}$.

3) Computation Cost: For a given frame structure $\mathcal{T}$, we can define the corresponding maximum *computation cost* during encoding, $S(\mathcal{T})$, by searching for the maximum number of encoded frames in $\mathcal{T}$ across all time instants:

$$S(\mathcal{T}) = \max_i \sum_{F_{i,j} \in \mathcal{T}} 1 \tag{1}$$

In Fig. 2(b), the computation cost of the redundant frame structure $\mathcal{T}$ shown is 4. This cost represents the maximum number of "encoders" that need to be operating simultaneously at any point in time (under the assumption that all frames corresponding to time i are encoded simultaneously.)

4) Transmission Cost: Given a structure $\mathcal{T}$ and associated schedule $\mathbf{G}$, we can define a corresponding transmission cost as the expected transmission rate given transition probabilities $\alpha_j(k)$'s and frame sizes $|F_{i,j}|$'s for all frames in $\mathcal{T}$. We assume the size of an I-frame for given view j is a constant: $|I_{i,j}| = r^I_j$. In contrast, size of a P-frame depends also on the frame used for prediction. Assuming $P_{i,j}$ is encoded using as a predictor $F_{i-1,k}$, the corresponding size is $|P_{i,j}| = r^P_j(k)$.

The expected transmission cost $C(\mathcal{T})$ is a weighted sum of the starting I-frames $|I_{0,k}|$'s, $1 \leq k \leq K$, and *recursive transmission costs* $c(I_{0,k})$'s, where $c(F_{i,j})$ is the expected future transmission cost given that a client has just viewed frame $F_{i,j}$. $c(F_{i,j})$ in turn can be written as a weighted sum of *recursive transition costs* $\Phi(F_{i,j}, k)$'s of possible transitions to other views k's in time instant $i+1$:

$$\begin{aligned} C(\mathcal{T}) &= \sum_k w_k \left(r^I_{0,k} + c(I_{0,k}) \right) \\ c(F_{i,j}) &= \sum_{k=\max(1,j-1)}^{\min(K,j+1)} \alpha_{i,j}(k) \Phi(F_{i,j}, k) \end{aligned} \tag{2}$$

The recursive transition cost $\Phi(F_{i,j}, k)$ is the expected transmission cost from $F_{i,j}$ to scheduled $F_{i+1,k}$ and beyond. For given $F_{i+1,k}$, it is a sum of size of $F_{i+1,k}$, and subsequent recursive transmission cost $c(F_{i+1,k})$ after transition:

$$\Phi(F_{i,j}, k) = |F_{i+1,k}| + c(F_{i+1,k}) \text{ where } F_{i,j} \overset{\mathbf{G}}{\Rightarrow} F_{i+1,k} \tag{3}$$

B. Optimization Problem Defined

We can now define the IMVS-live problem as follows: find a structure $\mathcal{T}$ in feasible space[4] Θ that possesses the smallest expected transmission cost $C(\mathcal{T})$ while a computation constraint $\bar{S}$ is observed:

$$\min_{\mathcal{T} \in \Theta} C(\mathcal{T}) \text{ s.t. } S(\mathcal{T}) \leq \bar{S} \tag{4}$$

[2]Given join session and view switching periods M' and M, respectively, where $M' > M$, we have $N = \left\lfloor \frac{M'}{M} \right\rfloor - 1$.

[3]If *rerouting* [3] is permitted, where an *alternative P-frame* $P_{i+1,k}$, whose predictor $F_{i,j}$ is not the currently observed frame by the client, is sent along with its predecessors $F_{i,j}$ etc for client decoding, then there are more than one logical schedule. In Fig. 2(b), $I_{0,2}$ can transit to $P_{1,1}$ if server sends both $I_{0,1}$ and $P_{1,1}$. We do not consider rerouting in this paper.

[4]The feasible space is the set of structures that enable all permissible transitions from frame $F_{i,j}$ to frame $F_{i+1,k}$, where transitions have been constrained depending on desirable application characteristics, e.g., j and k may be constrained to be neighboring views.

V. IMVS-LIVE ALGORITHM

We now present an algorithm that constructs a redundant frame structure $\mathcal{T}$ for the IMVS-live optimization problem presented in Section IV. The algorithm is greedy in nature, and hence the algorithmic complexity is comparatively low. In words, the algorithm constructs structure $\mathcal{T}$ and associated frame-to-frame schedule $\mathbf{G}$ from front to back, where at each time instant we iteratively identify a transition that would reap the largest decrease in transmission cost by branching to a new P-frame instead of an I-frame, until the available computation resource is depleted.

We first note that the transmission cost of a structure $\mathcal{T}$, $C(\mathcal{T})$, can be written as a function of *display probabilities* $q(F_{i,j})$'s—the probability that $F_{i,j}$ is selected for display:

$$C(\mathcal{T}) = \sum_{F_{i,j} \in \mathcal{T}} q(F_{i,j}) |F_{i,j}| \tag{5}$$

For given structure $\mathcal{T}$, $q(F_{i,j})$'s can be calculated from front to back as follows. First, we are given initial probabilities for starting I-frames $I_{0,k}$'s, w_j's. Second, a client will branch from $F_{i,j}$ to a P-frame $P_{i+1,k}$ with display probability $q(F_{i,j})\alpha_j(k)$. Finally, a client can branch to an I-frame $I_{i+1,k}$ from multiple frames in previous instant; the display probability of $I_{i+1,k}$ is the sum of all transition probabilities from these frames.

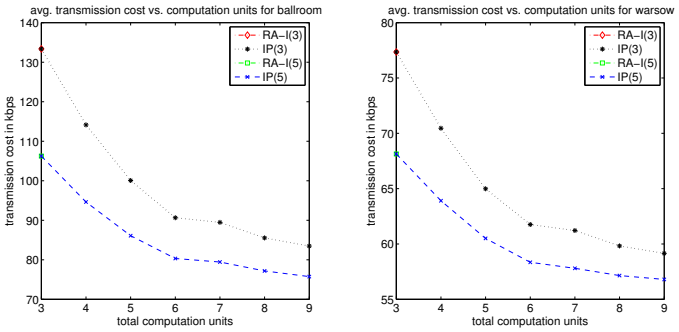
We can now present the algorithm in details as follow:

1) Initiate $\mathcal{T}$ with K I-frames $I_{0,k}$'s for instant $i := -1$.
2) $i := i + 1$. Create K I-frames $I_{i+1,k}$'s and schedule all feasible transitions to $I_{i+1,k}$'s. Calculate display probabilities $q(I_{i+1,k})$'s. Initiate computation unit $S := K$.
3) While $(S < \bar{S})$,
   a) Find a transition $F_{i,j} \stackrel{\mathbf{G}}{\Rightarrow} I_{i+1,k}$ s.t. if branches to a P-frame $P_{i+1,k}$ instead of I-frame $I_{i+1,k}$, it will decrease transmission cost maximally; i.e., $\max q(F_{i,j})(r_k^I - r_k^P(j))$. Create $P_{i+1,k}$. Schedule $F_{i,j}$ to branch to $P_{i+1,k}$ instead of $I_{i+1,k}$. $S := S + 1$.
   b) If the number of transitions to I-frame $I_{i+1,k}$ where the maximal transition was removed is now one, then replace $I_{i+1,k}$ with P-frame $P_{i+1,k}$.
4) if $(i < N - 1)$, goto step 2.

VI. EXPERIMENTS

For MV video data, we downloaded three consecutive views of the 100-frame `ballroom` sequence from [8] captured at 25fps and down-scaled to QCIF (176×144). In addition, we modified the game client of Internet game `warsow` [9] to render and capture a 100-frame sequence of three views simultaneously at 10fps, each displaced by 30^o. The motivation is to supply two very different data sets with different frame rates and camera distances.

For each sequence, we averaged the first five I-frames and P-frames predicted from previous frames of various views to generate r_j^I's and $r_j^P(k)$'s as inputs to our algorithm, using H.264 JM version 12.4 [10] as the default codec. For transition probabilities $\alpha_j(k)$'s, we assume frame $F_{i,1}$ remains at the same view $F_{i+1,1}$ with probability $1 - \alpha$, and transitions to neighboring views $F_{i+1,0}$ and $F_{i+1,2}$ with probability $\alpha/2$ each. Frame $F_{i,0}$ ($F_{i,2}$) transition to the single neighboring view $F_{i,1}$ with the same probability α. We assumed $\alpha = 0.1$, starting probabilities for each frame $I_{0,k}$ $w_k = 1/3$ for each of the three views, and switching instants $N = 9$.



(a) `ballroom` (b) `warsow`

Fig. 3. Tradeoffs between Transmission and Computation Costs

For a given fixed structure generated by our algorithm, we encoded the 100-frame test sequences into the structure M frames at a time and computed the transmission and computation costs. In Fig. 3, we see the performance of our algorithm (`IP`) for different switching period against the default random access I-frame insertion approach (`RA-I`). We see indeed that transmission and computation units can be traded off gracefully. In particular, at twice the computation, our proposed algorithm generated structures that reduced expected transmission costs by 32.9% and 20.1% for `ballroom` and `warsow` when $M = 3$, and by by 24.3% and 14.3% for `ballroom` and `warsow` when $M = 5$. We observe that our algorithm reaps the most gain when the cameras are spatially close, frame rates are high, and switch periods are small.

REFERENCES

[1] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Efficient prediction structures for multiview video coding," in *IEEE Transactions on Circuits and Systems for Video Technology*, November 2007, vol. 17, no.11, pp. 1461–1473.

[2] M. Flierl, A. Mavlankar, and B. Girod, "Motion and disparity compensated coding for multiview video," in *IEEE Transactions on Circuits and Systems for Video Technology*, November 2007, vol. 17, no.11, pp. 1474–1484.

[3] G. Cheung, A. Ortega, and N.-M. Cheung, "Generation of redundant coding structure for interactive multiview streaming," in *17th International Packet Video Workshop*, Seattle, WA, May 2009.

[4] N.-M. Cheung, A. Ortega, and G. Cheung, "Distributed source coding techniques for interactive multiview video streaming," in *27th Picture Coding Symposium*, Chicago, IL, May 2009.

[5] A. Jagmohan, A. Sehgal, and N. Ahuja, "Compression of lightfield rendered images using coset codes," in *Thirty-Seventh Asilomar Conference on Signals, Systems and Computers*, November 2003, vol. 1, pp. 830–834.

[6] Prashant Ramanathan and Bernd Girod, "Random access for compressed light fields using multiple representations," in *IEEE International Workshop on Multimedia Signal Processing*, Siena, Italy, September 2004.

[7] I. Bauermann and E. Steinbach, "RDTC optimized compression of image-based scene representation (part I): Modeling and theoretical analysis," in *IEEE Transactions on Image Processing*, May 2008, vol. 17, no.5, pp. 709–723.

[8] "Test sequences at MERL," ftp://ftp.merl.com/pub/avetro/mvc-testseq.

[9] "Warsow: a fast paced first person shooter game," http://www.warsow.net.

[10] "The TML project web-page and archive," http://kbc.cs.tu-berlin.de/˜stewe/vceg/.