

# OPTIMIZING FRAME STRUCTURE FOR INTERACTIVE MULTIVIEW VIDEO STREAMING WITH VIEW SYNTHESIS

Xiaoyu Xiu<sup>1</sup>, Gene Cheung<sup>2</sup>, Antonio Ortega<sup>3</sup>, and Jie Liang<sup>1</sup>

<sup>1</sup> Simon Fraser University, <sup>2</sup> National Institute of Informatics, <sup>3</sup> University of Southern California

## ABSTRACT

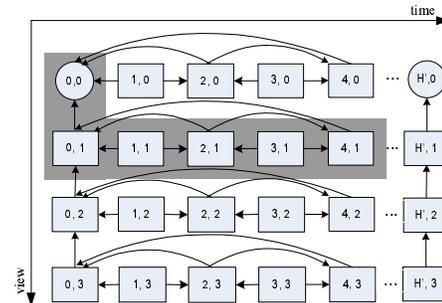
Traditional multiview video coding schemes compress all captured video frames exploiting all possible inter-view and temporal frame correlation for coding gain, creating complex inter-frame dependencies in the process. In contrast, interactive multiview video streaming (IMVS) demands data navigation flexibility in the frame structure design, so that server can send only a single periodically selected video view for decoding and display at client, saving transmission bandwidth. In this paper, we generalize previous IMVS frame structure optimization to allow a client to request an arbitrary virtual view; i.e., the server sends two adjacent coded views for the client to synthesize the desired virtual view. Since existing IMVS schemes transmit only one view at a time, they employ only cross-time prediction; i.e., the frame of previous time instant from which the client switches is used as predictor for the requested view. In our new scenario, two coded views are transmitted, thus within-time prediction can also be used, where the coded frame of one transmitted view is used to predict the frame of the other view of same time instant. Using I-frames, P-frames and Merge (M-) frames as building blocks, we formulate a Lagrangian problem to find the optimal frame structure for a desired storage/streaming rate tradeoff, with the right mixture of cross-time / within-time prediction types. Experiments show that for the same storage cost, the expected streaming rate of the proposed structure can be 40% lower than that of the I-frame-only structure, and 9% lower than that of the structure using M-frames but with cross-time prediction only.

**Index Terms**— Multiview video, video streaming, view interpolation / synthesis

## 1. INTRODUCTION

Multiview video refers to videos captured synchronously by multiple closely spaced cameras. For the sake of raw compression gain, traditional multiview video coding schemes (MVC) [1, 2] exploit inherent inter-view and temporal correlation of all captured video frames across time and view, resulting in complex inter-frame dependencies in the MVC frame structures. These complex inter-frame dependencies translate to a high transmission rate for applications such as *Interactive Multiview Video Streaming* (IMVS) [3, 4], where a client watches only a *single view* at a time, but can periodically request switches to neighboring views from server every  $H$  frames, as the single-view video is streamed and played back in time. This is because typical MVC frame structures are not optimized to provide sufficient decoding flexibility to support this view-switching interaction, and hence often multiple frames need to be transmitted just so a single desired frame can be correctly decoded.

As an illustration, Fig. 1 shows one MVC frame structure proposed in [1], where I-frames are periodically inserted every  $H'$  frames to permit some level of random access. In order to facilitate view switches every  $H$  frames, the structure in Fig. 1 can be generated with  $H'$  set to  $H$ . However, for a small desired view-switching

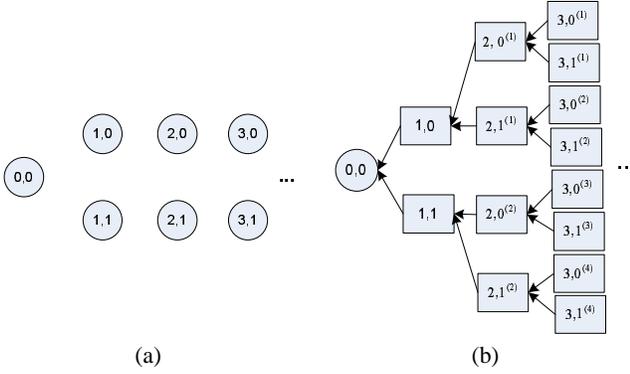


**Fig. 1.** Example of MVC frame structure, where circles and rectangles denote I- and P-frames, respectively.  $(i, j)$  denotes a frame at instant  $i$  of view  $j$ . Frames in the shaded region are transmitted to receiver to correctly decode and observe view 1.

period  $H$ , this leads to high transmission costs due to frequent I-frame insertion. Alternatively, one can first select a compression-efficient frame structure with  $H' \gg H$ , and then send to client all the frames required to enable decoding of frames in a single requested view after a view-switch. For instance, in order to switch from frame  $(2, 1)$  of time instant 2 and view 1 to frame  $(3, 2)$  of next instant 3 and neighboring view 2, server would send frames  $(0, 2)$ ,  $(2, 2)$ ,  $(3, 2)$  and  $(4, 2)$  to client, but only frame  $(3, 2)$  is displayed. Besides a large overhead in decoding complexity, this incurs a large transmission cost.

In contrast to MVC schemes focusing solely on efficient compression of all frames of all views, the goal of IMVS is to design structures at encoding time to optimally trade off expected transmission rate at stream time and storage required to store the structure, while enabling view switching. To provide intuition on the tradeoff between storage and transmission rate, consider the following two frame structures in Fig. 2 for two views and  $H = 1$ . The first structure employs I-frames at all possible view-switching points. That means no matter which sequence of frames in the structure (decoding path) a client was traversing when she switches views, she can correctly decode the I-frame of her requested view after the switch. However, this requires high transmission rate due to large sizes of I-frames.

The second structure deploys *redundant P-frames* at all view-switching points; at each decoding path, one P-frame is pre-encoded for each possible switched view, using the last frame in the decoding path as predictor. Although this structure has the minimum transmission cost, the storage required is prohibitive, due to large number of P-frames required. In [3, 4], using I-frames, redundant P-frames and *Merge* (M-) frames (a single frame that enables view-switching from multiple decoding paths but exploits correlation between target frame and frames in decoding paths for coding gain) as building blocks, optimal and heuristic algorithms were developed to achieve



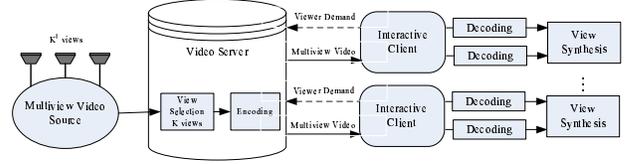
**Fig. 2.** Example of IMVS Frame structures for two views, where circles and rectangles denote I- and P-frames, respectively.  $(i, j)$  denotes a frame at instant  $i$  of view  $j$ . (a) structure with I-frames only; (b) structure with an initial I-frame and all P-frames.

different tradeoffs between those two extreme structures in terms of storage rate and expected streaming rate.

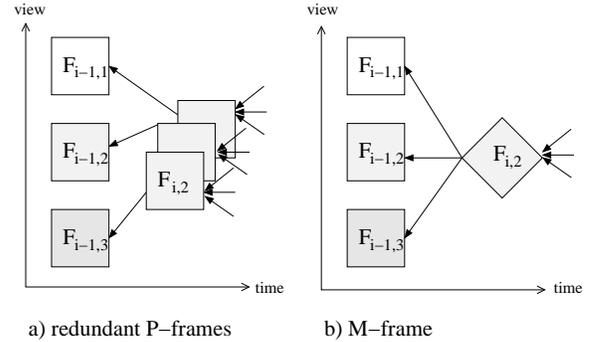
One shortcoming for IMVS is that available views for a client are limited by the few discrete number of camera-captured views pre-encoded at server, which means a view-switch can appear abrupt and unnatural to a viewer. In this paper, we introduce *arbitrary view switching*: in addition to camera-captured views, *virtual views* in-between captured views can also be requested by clients. Theoretically, arbitrary view switching offers viewers an *infinite* number of views, so that a view-switch can now take place between views as close/far as the user desires. A virtual view can be synthesized using images at two nearest captured views via *image-based rendering* (IBR) [5], or recently popular *depth-image-based rendering* (DIBR) [6]. In the latter case, both texture and depth images at captured viewpoints need to be available for view synthesis. Encoding both texture and depth images at multiple camera-captured viewpoints is commonly called the *video+depth* format [7]. Though we focus in this paper on IBR and encode texture images only, our proposed structure optimization can potentially handle video+depth format as well: we can use the same optimized structure to encode texture and depth images of respective viewpoints separately.

To enable arbitrary view switching but maintain reasonable workload at server, the server transmits *two* nearest coded views to the client as references for view synthesis of the requested virtual view at client. We call this system IMVS with View Synthesis (IMVS-VS); see Fig. 3 for an illustration. Our goal is to design an efficient pre-encoded frame structure of a multiview video sequence at the server to facilitate arbitrary view switching. A natural approach to enable arbitrary view switching in IMVS-VS is to use the same IMVS frame structure composed of I-frames, M-frames and *cross-time* predicted (CT) P-frames, where a P-frame is encoded using a frame in previous time instant as predictor. However, in IMVS-VS the server always transmits two frames from two neighboring views of the same instant, and those two frames typically exhibit high spatial correlation. We can hence achieve better performance by enabling *within-time* (WT) prediction also, where one transmitted frame can be predicted using the other transmitted frame as predictor, as done in MVC.

In this paper, using I-, P- and M-frames as building blocks, we formulate a Lagrangian problem to find optimal frame structures that enable arbitrary view switching in IMVS-VS. The crux of the optimization lies in finding the right mixture of the two prediction types



**Fig. 3.** System Overview of IMVS-VS System.



**Fig. 4.** Example of redundant P- and M-frames.

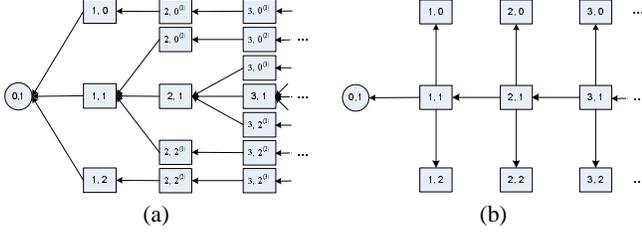
(CT and WT) for P-frames that offers the best storage / streaming rate tradeoff. Experimental results show that structures using appropriately added WT-predicted P-frames can lower streaming bitrate of IMVS-VS over I-frame-only structure by up to 40%, and over structure with M-frames but only CT-predicted P-frames by up to 9% for the same storage constraint.

The outline of the paper is as follows. Sec. 2 overviews our proposed IMVS-VS framework and frame structure. In Sec. 3, we formulate the problem of generating the optimal frame structure for IMVS-VS. A greedy algorithm is then developed in Sec. 4 to generate the structure. Experiment results and conclusion are given in Sec. 5 and Sec. 6, respectively.

## 2. FRAME STRUCTURE IN IMVS-VS

In our proposed IMVS-VS system, videos from  $K^0$  closely spaced cameras in a 1D array capture a scene of interest for later streaming and playback. Given  $K^0$  captured views, one can design a frame structure at encoding time to optimally trade off expected transmission rate at stream time and storage size of the entire structure. Note that given client has the ability to synthesize any intermediate view using images of two closest captured views via IBR or DIBR, one can choose a *subset of  $K$  coded views* from  $K^0$  captured views for coding at server instead, and still enables arbitrary view switching at client. Doing so would mean coding fewer total views, resulting in a better transmission rate / storage. However, it also results in higher synthesized view distortion, since the two closest captured views used for view synthesis are now further apart. We assume  $K$  of  $K^0$  views were pre-selected a priori based on minimum synthesized view quality requirement as specified by the application.

A frame structure is encoded offline using a redundant frame structure composed of I-, P- and M-frames. An M-frame is a single-frame representation, where upon a view-switch, the exact same decoded image can be correctly reconstructed no matter from which one of a set of known decoding paths a user is switching. Yet unlike an I-frame, an M-frame exploits correlation between frames in different decoding paths and the target frame for coding gain. Fig. 4 shows examples of redundant P-frames and M-frame. We see that



**Fig. 5.** Example frame structures of IMVS-VS for three coded views and  $H = 1$ . Circles and rectangles denote I- and P-frames, respectively.  $(i, j)$  denotes a frame at instant  $i$  of view  $j$ . (a) using CT prediction only; (b) using both CT and WT predictions.

unlike redundant P-frames in Fig. 4(a) that results in three different coded versions of frame  $F_{i,2}$  given three different predictors  $F_{i-1,1}$ ,  $F_{i-1,2}$  and  $F_{i-1,3}$ , M-frame reconstructs the same coded version no matter which one of the three possible predictors is available at decoder's buffer. Example implementations of M-frame includes H.264 SP-frames [8] and specially designed *distributed source coding* (DSC) frames for IMVS [9]. While our structure optimization applies for any implementation of M-frames, for the sake of simplicity, we will use only DSC frames designed in [9] as the chosen implementation of M-frame for the sequel.

A client can request video of an arbitrary viewpoint, and can switch viewpoint every  $H$  frames. To facilitate view synthesis at the client side, the server always transmits two neighboring coded views, and the client uses view interpolation to generate the virtual view [5]. The two boundary views, view 0 and view  $K^0 - 1$ , must be pre-selected as coded views to enable synthesis of any virtual view between view 0 and view  $K^0 - 1$ . In addition, the server encodes every  $H'$ -th frame of each coded view as an I-frame,  $H' \gg H$ , to permit some required level of random access.

## 2.1. Examples of IMVS-VS Structures

To illustrate the benefit of WT-predicted P-frames for IMVS-VS, Fig. 5 shows two example structures with three coded views and  $H = 1$ . Structure in Fig. 5(a) has an I-frame followed by successive CT-predicted P-frames. Only two P-frames need to be transmitted for any view-switch. In Fig. 5(b), WT prediction is used in addition to CT prediction, where after the first I-frame of view 1, other frames of view 1 are coded as CT-predicted P-frames by using the previous frame of view 1 as predictor. Pictures of view 0 and view 2 are coded as WT-predicted P-frames by using the frame of view 1 of same instant as predictor. This structure *also* only sends two P-frames for any possible view-switch. However, compared to Fig. 5(a), WT prediction can greatly reduce the number of coded frames, offering a much better storage / streaming rate tradeoff.

In the sequel, we call a particular coded version of an original captured image a *coded frame*. We use  $F_{i,j}$  to denote a coded frame at time instant  $i$  and view  $j$ . We assume a view switching model<sup>1</sup> where, after observing a virtual view  $x$ , the client can switch to a neighboring virtual view within a range of  $L$  for the next instant  $i + 1$ , i.e., a view position  $y \in [x - L, x + L]$  will be requested with continuous *view transition probability*  $\phi_i(y - x)$ .

<sup>1</sup>Though we present the interactive model for view-switching interval  $H = 1$ , it can be easily generalized to the case  $H > 1$  by representing  $F_{i,j}$  as  $H$  consecutive coded frames of the same view  $j$ .

## 3. PROBLEM FORMULATION

### 3.1. Optimal Transmission Schedule

Given a fixed number of coded views  $K$ , one can construct a redundant frame structure  $\mathcal{T}$  of the multiview sequence to enable arbitrary view switching for IMVS-VS.  $\mathcal{T}$  is redundant in the sense that an original captured image can be represented by multiple coded frames  $F_{i,j}^{(k)}$ 's. For example, in Fig. 5(a), captured image of instant 2 and view 0 is represented by two coded frames,  $F_{2,0}^{(1)}$  and  $F_{2,0}^{(2)}$ .

Depending on the decoding path traversed by the user, different coded frames representing the same original captured image will be transmitted. Hence a *transmission schedule*  $G$  associated with a given redundant structure  $\mathcal{T}$  is needed. Let  $\Xi_i$  be the pair of coded frames cached at the decoder at instant  $i$  when the user requests a virtual view  $y$  for instant  $i + 1$ . The schedule  $G$  dictates which pair of coded frames,  $\Xi_{i+1}$ , should be transmitted at instant  $i + 1$ , such that that virtual view  $y$  can be synthesized at decoder. In this paper, we denote a scheduled transmission to view  $y$  by  $G$  as:  $\Xi_i \xrightarrow{G,y} \Xi_{i+1}$ ,

### 3.2. Optimization Problem

After defining a schedule  $G$  for a given frame structure  $\mathcal{T}$ , we can now define the design of an optimal redundant frame structure as an optimization problem. We first present some necessary definitions, then formally define the problem.

#### 3.2.1. Definitions

A). *View Display Probability and Frame Transmission Probability*: For ease of discussion, we first define *view display probability*  $f_i(x, \Xi_i)$  as the probability that virtual view  $x$  is synthesized at decoder using coded frame pair  $\Xi_i$  at instant  $i$ . If we assume that all IMVS-VS sessions start from view  $K^0/2$  with probability one, then this probability could be computed recursively using view transition probability  $\phi_i(y - x)$  and schedule  $G$ :

$$f_0(y, \{\}) = \delta\left(y - \frac{K^0}{2}\right) \quad (1)$$

$$f_{i+1}(y, \Xi_{i+1}) = \sum_{\Xi_i | \Xi_i \xrightarrow{G,y} \Xi_{i+1}} \int f_i(x, \Xi_i) \phi_i(y - x) dx$$

In words, (1) states that  $f_{i+1}(y, \Xi_{i+1})$  is the sum of probability  $\phi_i(y - x)$  of virtual view  $x$  of instant  $i$  switching to view  $y$  of instant  $i + 1$ , scaled by probability  $f_i(x, \Xi_i)$  of  $x$  itself, given schedule  $G$  dictates coded frame pair  $\Xi_{i+1}$  is used for synthesis of view  $y$ .

Correspondingly, we define *frame transmission probability*  $q(F_{i,j})$  as the probability that a coded frame  $F_{i,j}$  is transmitted from server to decoder to synthesize virtual views, which can be calculated using defined view display probability (1):

$$q(F_{i,j}) = \sum_{\Xi_i | F_{i,j} \in \Xi_i} \int f_i(x, \Xi_i) dx \quad (2)$$

In words, the transmission probability of a coded frame  $F_{i,j}$  is the sum of probability  $f_i(x, \Xi_i)$  of synthesized view  $x$ , where view  $x$  is interpolated using coded frame pairs  $\Xi_i$ 's that include  $F_{i,j}$ .

B). *Storage Cost*: For a given frame structure  $\mathcal{T}$ , we can define the corresponding *storage cost* by simply adding up the sizes of all the coded frames  $F_{i,j}$ 's in  $\mathcal{T}$ , i.e.,

$$B(\mathcal{T}) = \sum_{F_{i,j} \in \mathcal{T}} |F_{i,j}| \quad (3)$$

C). *Transmission Cost*: Given a frame structure  $\mathcal{T}$  and associated schedule  $G$ , *transmission cost* is defined as the sum of the sizes of all the coded frames  $F_{i,j}$ 's in  $\mathcal{T}$ , scaled by the corresponding frame transmission probabilities  $q(F_{i,j})$ 's:

$$C(\mathcal{T}) = \sum_{F_{i,j} \in \mathcal{T}} q(F_{i,j})|F_{i,j}| \quad (4)$$

### 3.2.2. Optimization Problem Definition

We can now define the design of redundant frame structure for IMVS-VS as an optimization problem: given a fixed number of coded views, how to find a structure  $\mathcal{T}^*$  and associated schedule  $G^*$ , using a combination of I-, P- and M-frames, that minimizes the transmission cost  $C(\mathcal{T})$  while a storage constraint  $\bar{B}$  is observed:

$$\arg \min_{\mathcal{T}} C(\mathcal{T}) \quad s.t. \quad B(\mathcal{T}) \leq \bar{B} \quad (5)$$

Instead of the constrained problem in (5), we solve the corresponding unconstrained Lagrangian problem, *i.e.*,

$$\min_{\mathcal{T}} J(\mathcal{T}) = C(\mathcal{T}) + \lambda B(\mathcal{T}) = \sum_{F_{i,j} \in \mathcal{T}} (q(F_{i,j}) + \lambda)|F_{i,j}| \quad (6)$$

where  $\lambda$  is the Lagrangian multiplier.

From (6), we see that a captured image can be represented by a number of P-frames, either CT-predicted or WT-predicted, each having a comparatively small transmission cost  $q(P_{i,j}^{(h)})|P_{i,j}^{(h)}|$ , but all together comprising a large storage  $\sum_h |P_{i,j}^{(h)}|$ . When  $\lambda$  is small, the penalty on large storage is negligible and multiple P-frames are attractive. On the other hand, when  $\lambda$  is large, the penalty on large storage cost becomes expensive and one single representation of the picture as I- or M-frame with relatively large transmission cost but small storage is more preferable.

## 4. ALGORITHM DEVELOPMENT

We now present a greedy algorithm to generate a good frame structure for IMVS-VS problem defined in Sec. 3. We first introduce the concept of segment to facilitate the calculation of frame transmission probability  $q(F_{i,j})$ , then discuss the algorithm in details.

### 4.1. Discrete Segment Probability

It can be seen from (6) that to calculate the Lagrangian cost, we have to first calculate frame transmission probability  $q(F_{i,j})$  in (2) using multiple integrals of synthesized view display probability  $f_i(x, \Xi_i)$ . To simplify the calculation of  $f_i(x, \Xi_i)$  then  $q(F_{i,j})$ , we approximate in discrete domain instead. In particular, we divide the interval between two boundary coded views, 0 and  $K^0 - 1$ , into  $N$  evenly spaced *segments*. More specifically, segment  $s_m$  represents virtual view positions located in the range  $\left[\frac{K^0-1}{N}m, \frac{K^0-1}{N}(m+1)\right)$ , where  $m \in \mathcal{Z}^+$ ,  $0 \leq m \leq N - 1$ .

As similarly done in Sec. 2, we now define *segment transition probability*  $\alpha_i[n - m]$  as the probability that upon watching a view  $x$  inside segment  $s_m$  at instant  $i$ , a client requests virtual view  $y$  inside segment  $s_n$ . It is straightforward to derive the expression of  $\alpha_i[n - m]$  from  $\phi_i(y - x)$  as follows:

$$\alpha_i[n - m] = \int_{x \in s_m} \int_{y \in s_n} \phi_i(y - x) dy dx \quad (7)$$

Assuming all virtual views within one segment use the same coded frame pair for synthesis (true for large enough  $N$ ), we can define *segment display probability*  $q_i^s(m, \Xi_i)$  as the probability that a synthesized view in segment  $s_m$  is synthesized at the client using coded frame pair  $\Xi_i$  at instant  $i$ . Similar to (1), we can recursively compute this probability using  $\alpha_i[n - m]$ , *i.e.*,

$$\begin{aligned} q_0^s(m, \{\}) &= \delta(m - m^0) \\ q_{i+1}^s(n, \Xi_{i+1}) &= \sum_{\Xi_i | \Xi_i \xrightarrow{G_i^s} n} \sum_{s_m} q_i^s(m, \Xi_i) \alpha_i[n - m] \end{aligned} \quad (8)$$

where  $s_{m^0}$  is the segment which includes the starting virtual view  $K^0/2$  in (1). Note that transmission schedule  $G$  now dictates transmission in terms of segments instead of individual virtual views. Correspondingly, (2) is approximated by

$$q(F_{i,j}) \approx \sum_{\Xi_i | F_{i,j} \in \Xi_i} \sum_{s_m} q_i^s(m, \Xi_i) \quad (9)$$

We can see that (9) is the discrete domain equivalent expression of (2). It is clear that the larger the segment number  $N$  is, the closer (9) approximates its real value defined in (2).

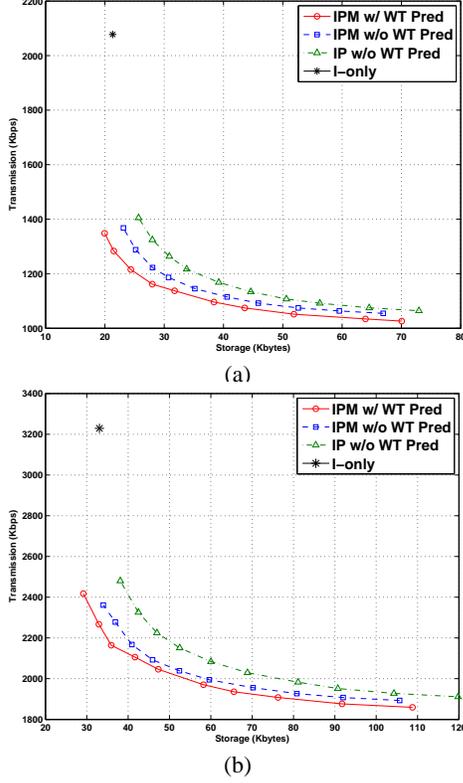
### 4.2. Greedy Structure Optimization

We now derive a greedy optimization algorithm to generate good frame structures, based on the optimization problem defined in Sec. 3.2. More precisely, we iteratively build one "slice" of the structure at each instant from front to back, *i.e.*, starting with two I-frames synthesizing view  $K^0/2$  at instant 0, we construct the local structure  $t_1$  and corresponding schedule  $g_1$  at instant 1, then  $t_2$  and  $g_2$  at instant 2 and so forth. At each switching instant  $i$ , the key question is: given the scheduled structure  $\mathcal{T}_{i-1}(G_{i-1})$  constructed up to instant  $i - 1$ , how to optimally construct coded frames of  $t_i$  and its schedule  $g_i$  at instant  $i$  to minimize (6) for a given  $\lambda$ .

To construct locally optimal structure slice  $t_i$  at instant  $i$ , we initialize slice  $t_i$  with one M-frame for each view. This initial slice has no redundant representation (one frame per captured image). Thus, it has minimum storage, while large sizes of M-frames will lead to a large transmission cost. Next, to methodically reduce transmission cost, we can incrementally add the most beneficial redundant P-frames one at a time, resulting in an increase in storage. We terminate when no more beneficial redundant P-frames can be added to further lower local Lagrangian cost.

In details, we describe the algorithm as follows. First, as initial solution for  $t_i$ , we construct an M-frame for each coded view  $j$  at instant  $i$ , where all viable view-switches to view  $j$  from coded frames  $F_{i-1,k}$ 's in  $\mathcal{T}_{i-1}$  could transition. We then determine the corresponding schedule  $g_i$  and compute the local Lagrangian cost in (6). Given the initial solution, we improve the local structure  $t_i$  by iteratively making augmentations: selecting a candidate from a set of structure augmentations that offers the largest decrease in local Lagrangian cost. The augmentations include:

- adding a new WT P-frame  $P_{i,j}^{WT}(F_{i,l})$  of view  $j$ , predicted from a coded frame  $F_{i,l}$  of view  $l$  of instant  $i$ .
- adding a new CT P-frame  $P_{i,j}^{CT}(F_{i-1,k})$  of view  $j$ , predicted from a coded frame  $F_{i-1,k}$  in  $\mathcal{T}_{i-1}$  of previous instant  $i - 1$ .



**Fig. 6.** Tradeoff between expected transmission and storage using different coding configurations: (a) Akko-Kayo; (b) Ballroom.

- selecting a different predictor  $F_{i,l}^T$  of the same instant  $i$  for a constructed WT P-frame  $P_{i,j}^{WT}$ .

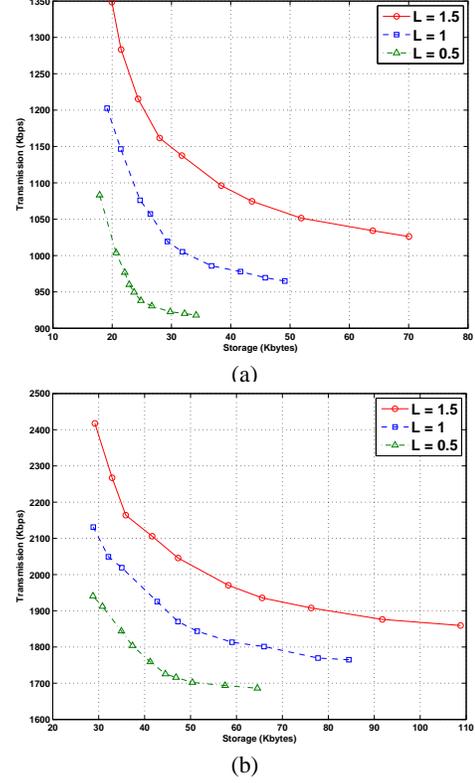
As shown above, the last augmentation does not increase the number of representations of a given coded view, while each of the first two increases the size by one P-frame. The above process repeats to find the most locally beneficial augmentation at each iteration, update the corresponding schedule and compute local Lagrangian cost, until no more Lagrangian cost reduction can be found. Note that after updating the local schedule at each iteration, it is possible that some coded frames in  $t_i$  are not used by any view-switch. In this case, those unused coded frames will be removed from the structure to save storage.

## 5. EXPERIMENTATION

### 5.1. Experimental Setup

We use H.263 tools to encode the first 90 frames of VGA size ( $640 \times 480$ ) sequence *Akko&Kayo* and *Ballroom* of 5 views ( $K^0 = 5$ ), at 30 frames per second. To generate data for DSC frames (our chosen implementation of M-frames), we use the algorithm in [9], developed using H.263 tools. We select quantization parameters such that I-, P- and DSC frames are reconstructed to the same quality (around 32dB). In addition, the random access period  $H'$  and switch period  $H$  are set to be 30 and 3, respectively. For view transition probability density function  $\phi_i(y-x)$ , in our experiment, we assume the following uniform distribution for simplicity:

$$\phi_i(y-x) = \begin{cases} \frac{1}{L+x} & 0 \leq x < L \\ \frac{1}{2L} & L \leq x \leq K^0 - 1 - L \\ \frac{1}{L+K^0-1-x} & K^0 - 1 - L < x \leq K^0 - 1 \end{cases} \quad (10)$$



**Fig. 7.** Tradeoff between expected transmission and storage using different view switch distance  $L$ : (a) Akko-Kayo; (b) Ballroom.

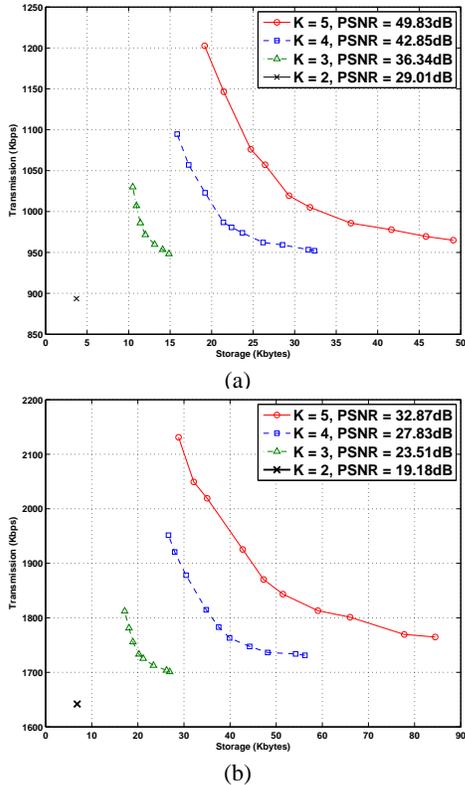
Correspondingly, segment transition probability  $\alpha_i(n-m)$  could be calculated from (7) as

$$\alpha_i(n-m) = \begin{cases} \frac{1}{\frac{LN}{K^0-1} + m + 1} & 0 \leq m < \frac{LN}{K^0-1} \\ \frac{1}{\frac{2LN}{K^0-1} + 1} & \frac{LN}{K^0-1} \leq m \leq N-1 - \frac{LN}{K^0-1} \\ \frac{1}{\frac{LN}{K^0-1} + N-m} & N-1 - \frac{LN}{K^0-1} < m \leq N-1 \end{cases} \quad (11)$$

We set the number of segments  $N$  to be 32, which we find to be a good tradeoff between computation complexity and performance. We also measure objective video quality in Peak Signal-to-noise Ratio (PSNR) of all the virtual views in each segment. Since no captured images are available for each virtual view, we use synthesized images, interpolated from the original uncompressed neighboring images, as reference to calculate PSNR.

### 5.2. Experimental Results

In Fig. 6, we compare the performance of frame structures generated using our proposed algorithm using I-, DSC and P-frames with WT prediction (IPM w/ WT), without WT prediction (IPM w/o WT), using I- and P-frames without WT prediction (IP w/o WT), and using only I frames (I-only). The view switching distance  $L$  is set to be 1.5. First, we observe that I-only had a single tradeoff point, because placing I-frames at all switching points results in no flexibility to trade off between storage and transmission rate. Second, for the same storage, IPM w/ WT offers lower transmission rates than I-only by up to 40% for *Akko&Kayo* and 30% for *Ballroom*, due to using both WT and DSC coded frames. Further, we observe that structures using WT prediction can offer a noticeable



**Fig. 8.** Tradeoff between expected transmission and storage using different number of coded views  $K$ : (a) Akko-Kayo; (b) Ballroom.

improvement over those using CT prediction only, with rate saving up to 9% for *Akko&Kayo* and 6% for *Ballroom*. Third, using DSC frames can generate better tradeoff points than using I-frames. The improvement is larger at stringent storage constraint, because DSC frames are more often used by the optimized structure to lower overall storage.

The tradeoff points of the proposed algorithm are plotted in Fig. 7, where different view-switch distance  $L$ 's are used. We can see that increasing  $L$  makes the performance deteriorate dramatically. This is intuitive; more coded frames are generally required to handle larger view-switch distance, resulting in larger storage.

In Fig. 8, we compare the tradeoff points when  $L = 1$  and different number of coded views  $K$  are used. More precisely, when  $K = 5$ , all the 5 views  $\{0, 1, 2, 3, 4\}$  are selected. When  $K = 4$ , there are in total three different view subsets to set up the system, namely,  $\{0, 2, 3, 4\}$ ,  $\{0, 1, 3, 4\}$  and  $\{0, 1, 2, 4\}$ . Therefore, the corresponding curve of  $K = 4$  is the average of the three cases. When  $K = 3$ , we choose the subset with equally spaced view positions, *i.e.*,  $\{0, 2, 4\}$ , to balance the synthesized quality of all the virtual view positions in the system. We can observe that in general fewer coded views  $K$  means a smaller transmission rate and storage, but a worse view synthesis quality. This is intuitive; more coded views means more coded frames need to be constructed, leading to worse storage / transmission tradeoff, while it also means the two neighboring coded views are closer to the requested virtual view, resulting in smaller synthesized distortion. Finally, we can find that there is only one tradeoff point when  $K = 2$ , independent of  $\lambda$ . This can also be easily explained; the structure with leading I-frames followed by all P-frames predicting from previous frames of the same view offers the smallest transmission and storage, regardless of  $\lambda$ .

Similar to Fig. 8, we conjecture that for the same viewing range between left-most and right-most cameras, view synthesis distortion could be further decreased by encoding more captured views  $K^0$  (more densely spaced cameras) at server, at the expense of the increase in both storage and transmission cost.

## 6. CONCLUSION

Unlike conventional multiview video coding (MVC) schemes that focus on compression of all frames, in this paper, we address the frame structure design problem for interactive multiview video streaming with view synthesis (IMVS-VS), where a user watches a single video view at a time, but can request view-switch from the server to an arbitrary view every  $H$  frames. Operationally, the server sends two closest coded views to the client so she can synthesize the desired intermediate view using image-based rendering (IBR) techniques. Using I-, P- and M-frames (Distributed Source Coding (DSC) frames are our chosen implementation of M-frames) as building blocks, we formulate the structure design problem as a Lagrangian minimization, and develop a greedy algorithm to generate good structures. The key observation is that unlike previous IMVS structures that relies on *cross-time* P-frames (each predictively coded using a frame of previous time instant in a decoding path), our structure optimization judiciously adds *within-time* P-frames as well, so that a frame in one transmitted view can be predictively coded using a frame in the other transmitted view of the same time instant, given server transmits two coded views for client's view synthesis. Experimental results demonstrate our optimized structures offer noticeably better streaming rate / storage tradeoffs than I-frame-only structures and previous IMVS structures.

## 7. REFERENCES

- [1] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Efficient prediction structures for multiview video coding," *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1461–1473, November 2007.
- [2] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, "Multi-view imaging and 3DTV," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 10–21, November 2007.
- [3] G. Cheung, A. Ortega, and N.-M. Cheung, "Generation of redundant coding structure for interactive multiview streaming," *Seventeenth International Packet Video Workshop, Seattle, WA, May 2009*.
- [4] G. Cheung, A. Ortega, and N.-M. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," *IEEE Transaction on Image Processing*, vol. 20, no. 3, pp. 744–761, March 2011.
- [5] X. Xiu and J. Liang, "Projective rectification-based view interpolation for multiview video coding and free viewpoint generation," *27th Picture Coding Symposium, Chicago, IL, May 2009*.
- [6] S.-T. Na, K.-J. Oh, C. Lee, and Y.-S. Ho, "Multi-view depth video coding using depth view synthesis," *IEEE International Symposium on Circuits and Systems*, Seattle, WA, May 2008.
- [7] P. Merkle, A. Smolic, K. Mueller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *IEEE International Conference on Image Processing*, San Antonio, TX, October 2007.
- [8] M. Karczewicz and R. Kurceren, "The SP- and SI-frames design for H.264/AVC," in *IEEE Transactions on Circuits and Systems for Video Technology*, July 2003, vol. 13, no. 7, pp. 637–644.
- [9] N.-M. Cheung, A. Ortega, and G. Cheung, "Distributed source coding techniques for interactive multiview video streaming," *27th Picture Coding Symposium, Chicago, IL, May 2009*.