Low-latency Error Control of H.264 Using SP-Frames and Streaming Agent Over Wireless Networks

Gene Cheung
Hewlett-Packard Laboratories Japan
3-8-13, Takaido-higashi
Suginami-ku, Tokyo, Japan 180–0022
Email: gene-cs.cheung@hp.com

Wai-tian Tan
Hewlett-Packard Laboratories Palo Alto
1501 Page Mill Rd
Palo Alto, CA 94304
Email: wai-tian.tan@hp.com

***Abstract*—A key challenge to low-latency wireless video streaming, where persistent retransmission is impractical, is error control. While SP-frame adaptation of H.264 has potential to mitigate error propagation, streaming server is often either situated too far to react in a timely fashion to client feedbacks, or too computationally constrained to perform the necessarily complex adaptation simultaneously for multiple clients in different sessions. In this paper, we present an innovative error control mechanism using SP-frames of H.264 and performed by a network intermediary for video streaming to a wireless client. Using an intermediary means it is more responsive to client feedbacks due to its close proximity, and it can offload computation complexity from the streaming server. Simulation shows that about 2 dB improvement in PSNR is achievable for video streaming with low latency requirements over traditional schemes using I and P-frames only.**

I. Introduction

Maintaining the quality of low-latency streaming video is challenging due to the large bandwidth usage and the strict requirement of continually delivering video frames on time. Even more difficult is low-latency streaming over wireless cellular networks, where rampant packet losses and considerable transmission delays further exacerbate the problem of error control. In such demanding streaming scenarios, only a small number of retransmissions are possible to recover a portion of lost packets. Schemes using forward error correction (FEC) incur additional channel encoding and decoding latency; moreover, for burst loss channels, significant interleaving is necessary to de-correlate channel losses for FEC to be effective, further increasing the end-to-end delivery delay.

At the application layer, one promising error control technique for H.264 video is the use of *SP-frame* [1], where a picture F_i can be identically reconstructed using only one of two encoded versions of SP-frame (*primary* and *secondary*), each predictively encoded using a different past frame (commonly previous frame F_{i-1} and a further back frame $F_{i-\delta}$) as reference. Error propagation at the client can be mitigated if either primary or secondary SP-frame is decoded on time. For a server to effectively utilize this feature, however, it must: i) react to losses reported by the client in a timely fashion; and, ii) to construct and send secondary SP-frames in real-time based on client feedbacks. The first task is difficult due to the large round trip delay in our wireless scenario, and the second is not always possible if the server needs to simultaneously serve multiple streaming clients in different sessions.

In a separate development, modern streaming systems are employing intermediaries [2] between servers and clients, as depicted in Fig. 1, to perform functions such as stream caching and relaying to improve streaming reliability and scalability. For wireless streaming, we have shown in our earlier work [3] that one such intermediary, the *Streaming Agent (SA)*, located at the junction of wired network and a wireless link, can effectively exploit any excess bandwidth in either sub-path to reduce loss rate. See Fig.2 for an illustration.

The SA is an ideal candidate to perform aforementioned SP-frame adaptation: i) its close proximity to the wireless client means it can respond in a more timely fashion to client feedbacks; and, ii) its real-time construction of secondary SP-

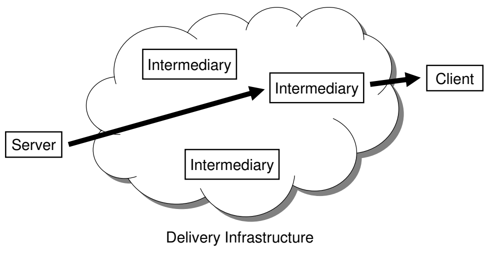

Fig. 1. Modern streaming systems employ intermediaries to improve scalability and reliability for large number of users.

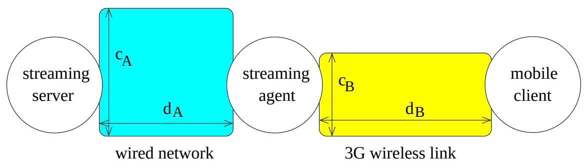

Fig. 2. A Streaming Agent can perform different number of retransmissions depending on bandwidth and delay of each sub-path.

frames means it offloads computational complexity from the server. Moreover, deploying SA as an intermediary means one can exploit fully the available bandwidth in wired network (c_A) and wireless link (c_B), while a simple server-client connection can only transmit at the minimum rate $\min(c_A, c_B)$.

In this paper, we propose a novel low-latency error control scheme using SP-frames of H.264 and performed by SA for wireless video streaming system. Specifically, we consider a streaming scenario where SA performs streaming optimization to dynamically decide whether to employ SP-frame adaptation, perform limited retransmission, or simply ignore packet losses. The rest of the paper is organized as follows. We present a discussion on related works in Section II, followed by an overview of our scheme in Section III. Associated optimization using SA is discussed in Section IV. Simulations and conclusions are given in Sections V and VI, respectively.

II. Related Work

Many video source coding techniques for improved error resilience is compatible with low-latency streaming. In particular, H.264 provides the Flexible Macroblock Order (FMO) and Arbitrary Slice Order (ASO) for error resilience [4]. These techniques attempt to limit error propagation *within* a frame, and cannot limit error propagation from frame to frame. Reference Picture Selection is another feature of H.263v2 and H.264 that can effectively stop temporal error propagation, but requires a live encoder and is not applicable to stored or pre-encoded content. Multiple description coding schemes are compatible with low-latency and burst loss channel only with the use of multiple disjoint delivery paths [5]. In this paper, our focus is on using SP-frames to control temporal error propagation; it is compatible with features such as FMO and ASO and applicable to stored content delivered over a single burst loss channel.

For SP-frames, most existing works focus on jointly selecting the many quantization parameters associated with primary and secondary SP-frames to optimize rate-distortion performance, and are not directly related to error resilience. In our earlier work [6] on using SP-frames for error resilience, we limit our consideration to the case of server-to-client transmission without involving a network intermediary, and only under moderate delay constraints of about one second. In this paper, we extend the earlier results to the more challenging scenario where delay constraint is more tight, and the intermediary, unlike the server, does not have access to the entire content at any time, and cannot transmit packets beforehand.

Works on network intermediaries [2], including our earlier work on Streaming Agent [3], focus on demonstrating the benefits of such entities using media agnostic channel coding or retransmission schemes. In contrast, our current work focus on a specific usage of such intermediaries in a fashion that is tightly coupled with the characteristics of SP-frames.

III. Overview: Using SP-frames for Low-latency Error Control

An overview of our scheme is illustrated in Fig. 3. A streaming server transmits video frames in sequence, with suitable number of retransmissions given each frame's playback deadline. Each frame arriving at an intermediary at a later time, and at a client at a yet later time. The low-latency client will display each video frame shortly after reception and discard frames that arrive past their display deadlines. Server uses video source with periodic (period Δ) video frames coded using primary SP-frames (shown in green). A key characteristic of primary SP-frames is that each allows the construction of a secondary SP-frame (shown in blue) that can *perfectly* reconstruct the same picture as the primary SP-frame, but from a different reference frame [1]. When irrecoverable packet losses (shown in red) occur in the intermediary-to-client sub-path, e.g., due to an expired deadline, the intermediary can wait for the arrival of the next primary SP-frame from server, then constructs and transmits the corresponding secondary SP-frame in place of the original primary SP-frame to the client to prevent error propagation. In our example, the intermediary is aware of the loss of frame 2 and 3 at the client and constructs and transmits the secondary SP-frame 4 that references the correctly received frame 1 instead of the primary SP-frame 4. A client can then repair its state with the secondary SP-frame 4, and the intermediary reverts back to normal operation.

Similarly, if a packet expires its client playback deadline in the server-to-intermediary sub-path, the intermediary can also wait for the next SP-frame, then construct and send the corresponding secondary SP-frame to the client to restore decoding state. In the example of Fig. 3, before encoding a secondary SP-frame, the target frame 4 needs to be correctly reconstructed, which requires reception and decoding of the first four frames. Nevertheless, for the purpose of constructing the secondary SP-frame, it is only necessary that all four frames are delivered by the deadline of frame 4, *even though* they may individually be too late for display purpose. This allows extra time for retransmissions of lost packets on the server-to-intermediary sub-path: frame 1 to 4 only need to arrive by deadline of frame 4 for construction of secondary SP-frame 4.

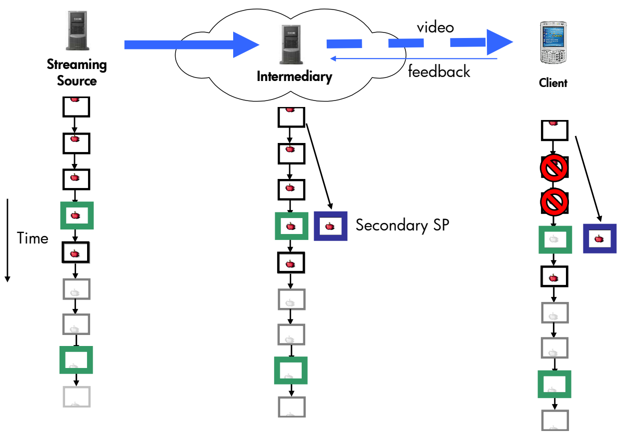

Fig. 3. On demand construction of secondary SP-frames is a low-latency error control mechanism to limit error propagation. The scheme is shown being performed in a network intermediary from stored compressed video.

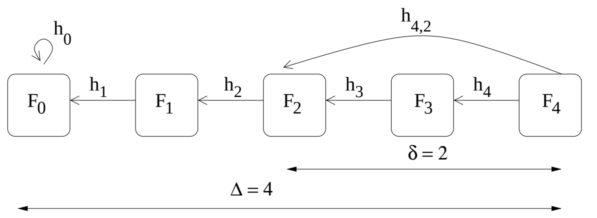

Fig. 4. Graphical representation of assumed source model with size of video frames represented in number of packets h.

The key benefits of the scheme includes: (1) low-latency, as minimal retransmissions and no interleaving are involved, (2) efficient elimination of error propagation due to late arriving frames, and (3) bandwidth efficiency. Even though a secondary SP-frame can be large, it is transmitted only on demand when losses occur, and can be smaller than the sum sizes of frames needed to repair decoder state (frames 2, 3 and 4 in Fig. 3). On the negative side, the lost frames (frames 2 and 3) remains corrupted, and the use of primary SP-frames adds a minor bit-rate overhead. In the remaining sections, we evaluate the balance of the pros and cons in a streaming scenario involving a streaming agent, where limited retransmission is performed between the two sub-paths given a delay constraint.

IV. SA-based Optimized Streaming using SP-frames

In our system in Fig.3, little intelligence is assumed for the streaming server or client besides the ability to initiate and respond to retransmission requests. The server simply streams video frames according to their presentation time, and the task of deciding whether and when to construct secondary SP-frames, and from which reference frame, is determined by our network intermediary—the Streaming Agent. A high-level overview of our scheme is already given in the previous section. In this section, we further discuss how to perform optimized streaming at the Streaming Agent. We will discuss the network and source models assumed in this paper, introduce relevant notations and show how appropriate objective functions can be computed, and optimization performed. In particular, we will discuss how to perform an offline optimization to determine primary frame frequency Δ in the video source before streaming commences in Section IV-C, and how to perform online optimization to decide when and how secondary SP frame should be used in Section IV-D. Simulation results using these assumptions and optimization will be presented later in Section V.

A. Source Model for SP-Frames

We assume a constant-frame-rate video source of N frames, $F_0, \dots F_{N-1}$, where F_0 is intra-coded, and frames F_Δ, $F_{2\Delta}$, and so forth are coded using SP-frame, and the rest are coded using P-frames. Frames $\left\{F_{j\Delta+1}, \dots, F_{(j+1)\Delta}\right\}$ are referred to as group j. Compressed frame of F_i is broken into h_i packets, each of which is no larger than a given *MTU* bytes. F_i must be delivered to the client by playback deadline T_i; we assume streaming session starts at time 0. The number of packets for the secondary SP-frame $F_{(j+1)\Delta}$ of group j using $F_{(j+1)\Delta-\delta_j}$ as reference is denoted by $h_{(j+1)\Delta,(j+1)\Delta-\delta_j}$. Fig.4 shows an example of our source model with $\Delta = 4$ and $\delta = 2$.

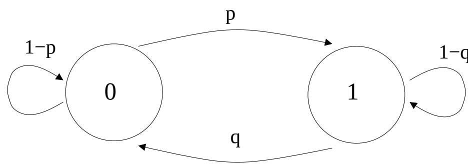

Fig. 5. Gilbert Loss Model

B. Network Model & Basic Definitions

We first assume wired network and wireless link have average bandwidth c_A and c_B and average forward-trip delay d_A and d_B, respectively. c_A can be obtained, for example, using an equation-based congestion avoidance algorithm called TCP-friendly Rate Control (TFRC) [7] based on observable end-to-end wired network statistics; c_B can be obtained during wireless session negotiation and setup between base-station and the wireless client [8]. We will use subscript A and B on various quantities throughout to denote wired network and wireless link, respectively. Given bandwidth c_A (c_B), a transmission duration t maps to an integer number of *transmission opportunities* $o_A(t)$ ($o_B(t)$) for fixed packet size u; more precisely, the numbers of packets that can be transmitted across the wired network and wireless link in duration t are respectively:

$$\begin{aligned} o_A(t) &= \left\lfloor \frac{c_A(t-d_A)}{u} \right\rfloor \\ o_B(t) &= \left\lfloor \frac{c_B(t-d_B)}{u} \right\rfloor \end{aligned} \tag{1}$$

Figure 2 illustrates the typical case where the wired network has larger bandwidth ($c_A > c_B$) and smaller transmission delay ($d_A < d_B$) than the wireless counterpart. In the simulation experiment, each packet experiences independent and identically distributed (iid) random variable delays γ_A and γ_B in wired network and wireless link respectively, each with its own shifted Gamma distribution $\Gamma_A(\gamma_A)$ and $\Gamma_B(\gamma_B)$.

We model the loss process in each of the wired network and wireless link as a discrete-time two-state Markov model (Gilbert model), with parameters (p_A, q_A) and (p_B, q_B) respectively. See Figure 5 for an illustration, where state 0 (1) implies a successfully (unsuccessfully) delivered packet.

Given a Gilbert model with parameters p and q, we can define the following definitions as done in [6]. Let $p(i)$, $i \geq 0$, be the probability of having *exactly* i consecutive correctly delivered packets between two lost packets, following an observed lost packet, i.e., $p(i) = \Pr(0^i 1|1)$. Let $P(i)$ be the probability of having *at least* i consecutive correctly delivered packets, following an observed lost packet, i.e., $P(i) = \Pr(0^i|1)$. $p(i)$ and $P(i)$ can be written as:

$$p(i) = \begin{cases} 1-q & \text{if } i=0 \\ q(1-p)^{i-1}p & \text{o.w.} \end{cases} \tag{2}$$

$$P(i) = \begin{cases} 1 & \text{if } i=0 \\ q(1-p)^{i-1} & \text{o.w.} \end{cases} \tag{3}$$

$q(i)$ and $Q(i)$ are complementarily defined functions; $q(i) = \Pr(1^i 0|0)$ and $Q(i) = \Pr(1^i|0)$.

We next define $R(m,n)$ as the probability that there are *exactly* m lost packets in n packets, following an observed lost packet. It can be expressed recursively as:

$$R(m,n) = \begin{cases} P(n) & \text{for } m=0 \text{ and } n \geq 0 \\ \sum_{i=0}^{n-m} p(i) R(m-1, n-i-1) & \text{for } 1 \leq m \leq n \end{cases} \tag{4}$$

We additionally define $r(m,n)$ as the probability that there are *exactly* m loss packets in n packets *between* two lost packets, following an observed lost packet. $r(m,n)$ can be similarly expressed recursively. Finally, we define $\bar{r}(m,n)$ as the probability that there are *exactly* m lost packets in n packets, following an observed lost packet and preceding a successfully received packet:

$$\bar{r}(m,n) = R(m,n) - r(m,n) \tag{5}$$

As counterpart to $R(m,n)$, $S(m,n)$ is the probability that there are *exactly* m received packets in n packets, following an observed received packet. $s(m,n)$ and $\bar{s}(m,n)$ are similar counterparts to $r(m,n)$ and $\bar{r}(m,n)$. Definitions of $S(m,n)$, $s(m,n)$ and $\bar{s}(m,n)$ are identical to their respective counterparts, with $Q(n)$ and $q(n)$ in place of $P(n)$ and $p(n)$.

C. Offline Optimization

We use offline optimization to determine the optimal primary SP-frame frequency Δ. We first describe the objective function used, then the algorithm to find the optimal Δ.

1) Objective Function: Suppose we fix the set of frames in the sequence we will send *a priori*: for each SP-frame, we select one of either primary or secondary version for fixed Δ; for P-frames, we forgo P-frames that a selected secondary frame references over (for example, F_3 for selected secondary F_4 in Fig. 4). Let D_i be the successful *decoding* probability of F_i. Selected frames leads to a set of Δ and δ_j's, and the offline objective value V_{off}, the expected number of timely decoded frames at the client, can be written as:

$$V_{off} = D_0 + \sum_{j=0}^{\lfloor \frac{N}{\Delta} \rfloor - 1} \left(\sum_{i=1}^{\Delta - \delta_j} D_{j\Delta+i} + D_{(j+1)\Delta} \right) \tag{6}$$

Let L_i be the successful *timely delivery* probability of F_i. D_i and L_i bear the following relation:

$$D_i = \begin{cases} L_i & \text{if } F_i \text{ is I-frame} \\ L_i D_{i-1} & \text{if } F_i \text{ is P-frame or primary SP} \\ L_i D_{i-\delta} & \text{if } F_i \text{ is secondary SP} \end{cases} \tag{7}$$

For the first I-frame, F_1 is successfully decoded iff it is correctly delivered from server to SA in exactly n packet transmissions, then correctly delivered from SA to client in remaining duration $T_1 - o_A^{-1}(n)$, where $o_A^{-1}(n) = \frac{nu}{c_A} + d_A$, for $0 \leq n \leq o_A(T_1)$. We can write:

$$\begin{aligned} L_1 &= \sum_{n=1}^{o_A(T_1)} L_{1,A}(n)\ L_{1,B}(o_B\left(T_1 - o_A^{-1}(n)\right)) \\ L_{1,A}(n) &= \pi_A \bar{r}_A(n-h_0, n-1) + (1-\pi_A) s_A(h_0 - 1, n-1) \\ L_{1,B}(m) &= \sum_{k=0}^{m-h_0} \pi_B R_B(k,m) + (1-\pi_B) S_B(h_0 + k, m) \end{aligned} \tag{8}$$

where $L_{1,A}(n)$ is the probability of F_1 being successfully delivered from server to SA in *exactly* n opportunities, and $L_{1,B}(m)$ is the probability of F_1 being correctly delivered from SA to client in *at most* m opportunities.

For P-frame or primary SP-frame $F_i, i \geq 1$, the calculation of L_i is more involved, due to uncertainty in the duration required for the successful delivery of previous frames F_1 to F_{i-1}. Let $P_i(\phi)$ be the probability that duration ϕ is available for F_i, i.e., server can begin transmission of F_i at time $T_i - \phi$ upon completing delivery of F_1 to F_{i-1} to SA. Similarly, let $Q_i(\zeta)$ be the probability that SA can begin transmission of F_i at time $T_i - \zeta$ upon completing delivery of F_1 to F_{i-1} to the client. L_i can then be analyzed as follows. Suppose duration ϕ is available to deliver F_i from server to client. If it takes n opportunities to send h_i packets from server to SA—with probability $L_{i,A}(n)$, that would leave duration $\phi - o_A^{-1}(n)$ for SA to send F_i to client. Upon F_i's arrival at SA, there are two cases: i) SA is ready and waiting at time $T_i - (\phi - o_A^{-1}(n))$, in which case the full duration $\phi - o_A^{-1}(n)$ is available for delivery of F_i from SA to client, or, ii) SA is still busy sending previous frames, in which case an even smaller duration is available for delivery of F_i. We can now write L_i as follows:

$$\begin{aligned} L_i = & \int_0^{T_i} P_i(\phi) \sum_{n=1}^{o_A(\phi)} L_{i,A}(n) \left[L_{i,B}\left(o_B\left(\phi - o_A^{-1}(n) \right) \right) * \right. \\ & \left. \int_{\phi - o_A^{-1}(n)}^{T_i} Q_i(\zeta) d\zeta + \int_0^{\phi - o_A^{-1}(n)} Q_i(\zeta)\ L_{i,B}(o_B(\zeta))\ d\zeta \right] d\phi \end{aligned} \tag{9}$$

In contrast, $L_{i,A}(n)$ and $L_{i,B}(n)$ are simpler because transmission of F_i at server and at SA always starts when the delivery of F_{i-1} is just successfully completed. That means the last packet transmitted for F_{i-1} is successful, and hence the channel is in good state 0:

$$\begin{aligned} L_{i,A}(n) &= s_A(h_i - 1, n-1) \\ L_{i,B}(n) &= \sum_{j=0}^{n-h_i} S_B(h_i + j, n) \end{aligned} \tag{10}$$

The crux is in evaluating $P_i(\phi)$ and $Q_i(\zeta)$ for F_i. Similarly done in [6], we approximate them as discrete functions: $P_i[n] = P_i(\frac{nu}{c_A})$ and $Q_i[m] = Q_i(\frac{mu}{c_B})$, for $n, m \in \mathcal{I}$. In words, $P_i[n]$ ($Q_i[m]$) is the probability that there are n (m) transmission opportunities for F_i from server (from SA) to client. We calculate $P_i[n]$ from previous F_{i-1}'s $P_{i-1}[n]$. First, F_i receives fresh transmission opportunities of $x_i =$

$o_A(T_i - T_{i-1} + d_A)$ relative to F_{i-1} due to its later playback deadline. d_A is added to avoid double counting transmission delay. Second, F_i inherits opportunities not expended by the delivery of F_{i-1}. More precisely, F_i will receive n leftover opportunities if F_{i-1} uses only $k-n$ attempts for h_{i-1} packets out of a budget of k opportunities. Summing this for all budget sizes, we get:

$$\begin{aligned} P_0[n] &= \begin{cases} 1 & \text{if } n = o_A(T_0) \\ 0 & \text{o.w.} \end{cases} \\ \hat{P}_i[n + x_i] &= \sum_k P_{i-1}[k]\ s_A(h_{i-1} - 1, k - n - 1) \\ P_i[n] &= \frac{1}{\bar{P}_i}\ \hat{P}_i[n] \end{aligned} \tag{11}$$

where $\bar{P}_i = \sum_n \hat{P}_i[n]$ is needed for normalization.

$Q_i[m]$ can be derived similarly. Fresh transmission opportunities $y_i = o_B(T_i - T_{i-1} + d_B)$ is added for the delivery of F_i relative to F_{i-1}. If delivery of F_{i-1} from server to SA is completed by time $T_{i-1} - o_A^{-1}(n)$, then duration $o_A^{-1}(n)$ is available for SA to deliver F_{i-1} to client. Again, there are two cases: i) $Q_i^{(1)}(n,m)$'s: SA is ready and waiting at time $T_{i-1} - o_A^{-1}(n)$, or, ii) $Q_i^{(2)}(n,m)$'s: SA is still busy sending previous frames. Considering both cases, we can write:

$$\begin{aligned} Q_0[m] &= \begin{cases} 1 & \text{if } m = o_B(T_0) \\ 0 & \text{o.w.} \end{cases} \\ \hat{Q}_i[m + y_i] &= \sum_n P_i[n + x_i] \left[Q_i^{(1)}(n,m) + Q_i^{(2)}(n,m) \right] \\ Q_i^{(1)}(n,m) &= s_B\left(h_{i-1} - 1, o_B\left(o_A^{-1}(n)\right) - m - 1\right) * \sum_{k=o_B\left(o_A^{-1}(n)\right)}^{o_B(T_{i-1})} Q_{i-1}[k] \\ Q_i^{(2)}(n,m) &= \sum_{k=0}^{o_B\left(o_A^{-1}(n)\right)-1} Q_{i-1}[k]\ s_B(h_{i-1} - 1, k - m - 1) \\ Q_i[m] &= \frac{1}{\bar{Q}_i}\ \hat{Q}_i[m] \end{aligned} \tag{12}$$

Given $P_i[n]$ and $Q_i[m]$, we can rewrite (9) more simply:

$$\begin{aligned} L_i = & \sum_{k=0}^{o_A(T_i)} P_i[k] \sum_{n=1}^{k} L_{i,A}(n) \left[L_{i,B}\left(o_B\left(o_A^{-1}(k-n)\right)\right) * \right. \\ & \left. \sum_{m=o_B\left(o_A^{-1}(k-n)\right)}^{o_B(T_i)} Q_i[m] + \sum_{m=0}^{o_B\left(o_A^{-1}(k-n)\right)-1} Q_i[m] L_{i,B}(m) \right] \end{aligned} \tag{13}$$

Given δ_j's, $L_{(j+1)\Delta}$ for secondary SP-frame $F_{(j+1)\Delta}$ of group j is similarly calculated with the following modifications:

- $P_{(j+1)\Delta}$ derives from $P_{(j+1)\Delta-\delta_j}$ instead of $P_{(j+1)\Delta-1}$.
- $P_{(j+1)\Delta}$ receives fresh opportunities $o_A(T_{(j+1)\Delta} - T_{(j+1)\Delta-\delta_j} + d_A)$.
- Transmission payload from server to SA for $F_{(j+1)\Delta}$ — h_i in $L_{i,A}(n)$ of (10) — is $\sum_{k=j\Delta-\delta_j}^{(j+1)\Delta} h_k$.
- $Q_{(j+1)\Delta}$ derives from $Q_{(j+1)\Delta-\delta}$ instead of $Q_{(j+1)\Delta}$.
- $Q_{(j+1)\Delta}$ receives fresh opportunities $o_B(T_{(j+1)\Delta} - T_{(j+1)\Delta-\delta_j} + d_B)$.
- Transmission payload from SA to client for $F_{(j+1)\Delta}$ — h_i in $L_{i,B}(n)$ of (10) — is $h_{(j+1)\Delta,(j+1)\Delta-\delta_j}$.

The payload changes reflect, respectively, the combined size of P-frames and primary SP-frame needed from server to SA before secondary SP-frame construction, and the size of the secondary SP-frame from SA to client.

2) Optimization Procedure: Having described the evaluation of objective (6), for a given Δ, we find the locally optimal set of δ_j's that maximizes V_{off} by adjusting one δ_j at a time, until the set converges. We compare the resulting objective value with other values obtained using other Δ settings. The optimal Δ is simply one with a corresponding set of δ_j's that maximizes (6).

D. Online Optimization

Unlike the offline optimization, the online optimization needs to be simple and efficient enough to be performed in real-time by SA. As such, both the objective function and the algorithm are essentially simplified versions of their offline counterparts.

1) Objective Function: Suppose $F_{j\Delta+i}$, $i < \Delta$, has just arrived at SA at time τ. The online objective function we chose is the expected number of timely decoded frames at the client for the remaining frames in this group: $F_{j\Delta+i}, \ldots, F_{(j+1)\Delta}$. Depending on the value of δ_j, the objective is evaluated one of two ways: if $i \leq (j+1)\Delta - \delta_j$, then $F_{j\Delta+i}$ up to $F_{(j+1)\Delta-\delta_j}$ P-frames and secondary SP-frame $F_{(j+1)\Delta}$ are included in this sub-group. Otherwise, only SP-frame $F_{(j+1)\Delta}$ is included. We write the objective V_{on}:

$$V_{on} = \begin{cases} \sum_{k=i}^{\Delta-\delta_j} D_{j\Delta+k} + D_{(j+1)\Delta}(\delta_j) & \text{if } i \leq (j+1)\Delta - \delta_j \\ D_{(j+1)\Delta}(\delta_j) & \text{o.w.} \end{cases} \tag{14}$$

where D_i and L_i are related as in (7). $D_a(b)$ is the timely decoded probability of SP-frame a if F_{a-b} is used as a reference. Timely delivery probability L_i is defined as follows:

$$L_k = \begin{cases} \sum_{m=0}^{o_B(T_k-\tau)-h_k} S_B(h_k + m, o_B(T_k - \tau)) & \text{if } k = i \\ \sum_n \Theta_k[n] \sum_{m=0}^{n-h_k} S_B(h_k + m, n) & \text{o.w.} \end{cases} \tag{15}$$

where $\Theta_k[m]$ can be updated from $\Theta_k[m-1]$ in a similar way as (11).

2) Optimization Procedure: The online optimization is simply to determine δ_j that would maximize V_{on}. If $\delta_j \leq (j+1)\Delta - i$, then we send $F_{j\Delta+i}$ as usual. Otherwise, we wait to construct and transmit secondary SP-frame $F_{(j+1)\Delta}$ after frames $F_{(j+1)\Delta-\delta_j}$ up to $F_{(j+1)\Delta}$ are all received at SA.

V. Results

In this section, we present simulation results using network topology of Fig.3 and optimization in Section IV. Our simulation results are divided into three parts. First, we look at the analytical performance trade-off of using different values of Δ. This is useful as part of an offline algorithm to determine Δ for encoding video content. Second, we look at simulation scenarios where the delay tolerance is medium ($700ms$ to

TABLE I
NETWORK PARAMETERS FOR SIMULATION

Wired Network		Wireless Link	
(p_A, q_A)	$(0.025087, 0.3333)$	(p_A, q_A)	$(0.027778, 0.25)$
κ_A	$20ms$	κ_B	$60ms$
(α_A, λ_A)	$(2, 0.1)$	(α_B, λ_B)	$(4, 0.2)$

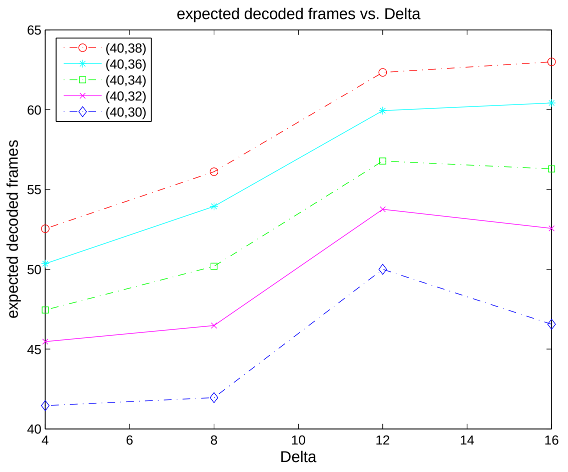

Fig. 6. Offline Optimization for `sean` for different Combinations of Wired / Wireless Bandwidths

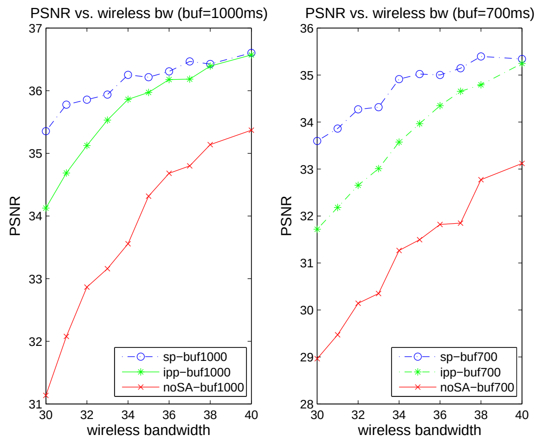

Fig. 7. PSNR of *Optimized-SP*, *Simple* and *Default* under wireless bandwidth constraint for `sean` at different client delay tolerance.

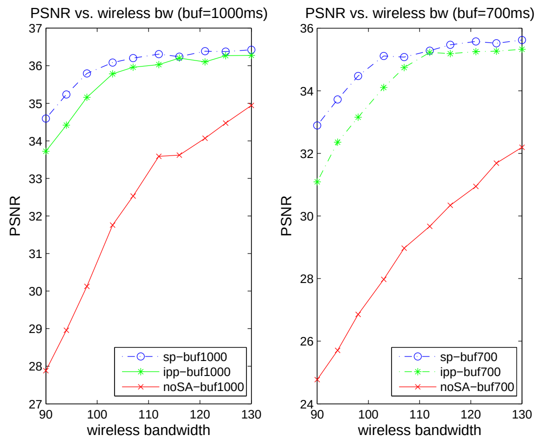

Fig. 8. PSNR of *Optimized-SP*, *Simple* and *Default* under wireless bandwidth constraint for `foreman` at different client delay tolerance.

$1000ms$) but the bandwidth of the SA-to-client sub-path is small. Finally, we look at the complementary case when delay tolerance is low (less than $500ms$) but the bandwidths in both sub-paths are plentiful.

For video source, we used 100-frame QCIF (176×144) video sequences of 10 frames per second. Each compressed video frame was broken into one or more packets of no more than 1500 bytes. The streaming server simply transmitted each packet according to its presentation time. We compared three schemes. In the *Default* scheme, only I and P-frames were stored in the server and no SA was used. Similarly in the *Simple* scheme, no SP-frames were used, but SA could request the server to retransmit lost packets in the server-to-SA sub-path, and could retransmit lost packets in the SA-to-client sub-path. In the *Optimized-SP* scheme, primary frames were inserted into the source at frequency Δ, and SA could perform retransmissions as in the *Simple* scheme as well as dynamically decide whether to construct and transmit a secondary SP-frame according to the online algorithm described in Section IV. The client discarded packets that were late for display purpose, and provided feedback regarding packet reception or loss.

For the first two parts of the experiment, the network parameters for the simulations are given in Table I, where p and q are parameters for the Gilbert loss model, and κ, α and λ are the parameters for the shifted-Gamma distributed delay model. They correspond to a packet loss rate of 0.07 and 0.10, burst length of 3 and 4, and average transmission delay of $40ms$ and $80ms$ for the wired network and wireless link, respectively.

A. Offline Optimization

The offline analytical results are shown in Figure 6. The client playback buffer was set to $1000ms$. Five plots of expected decoded frames vs. SP-frame frequency Δ for the `sean` sequence are shown for various combinations of wired network and wireless link bandwidths in $kbps$. We see that the objective V_{off} improved to an optimum as Δ increased, then trailed off as Δ further increased. Δ that yielded the optimal V_{off} was the frequency at which we inserted primary SP-frames into the video at the server. It is shown that for very large bandwidths, it is better to use the largest Δ, as most packets likely get to the client safely, and an error-resilient but less compression-efficient video representation using a small Δ is not necessary.

B. Online Optimization: Medium Delay, Small Bandwidths

We compared the performance of *Optimized-SP*, *Simple* and *Default* at different wireless bandwidths by running 3000 simulation runs each. The results for the `sean` and `foreman` sequence in Peak Signal-to-Noise Ratio (PSNR) are shown in Fig. 7 and Fig. 8, respectively. The bandwidth for the server-SA sub-path was fixed at $40kbps$ and $130kbps$ for the two sequences respectively. The average rate for the sequences were $27.8kbps$ and $82.1kbps$. The client was configured with a $700ms$ or $1000ms$ buffer delay which permited a limited number of retransmissions. Within this window, the burst-loss nature of the channel together with the variable-rate nature of video source meant that there could be temporary losses. We see that in Fig. 7 and Fig. 8, *Optimized-SP* always outperformed *Simple* and *Default*: up to $1.9dB$ and $4.6dB$ for `sean` and $1.8dB$ and $8.1dB$ for `foreman` when buffer delay was $700ms$. This indicates the effectiveness of the proposed scheme using SP-frames over traditional scheme using I and P-frames only.

C. *Online Optimization: Low Delay, Large Bandwidths*

Next, we compared the transmission of *Optimized-SP* and *Simple* at small display deadlines (low delay) and when bandwidths were plentiful. The propagation delay for each of the sub-path was set to $50ms$, and bandwidths in both links were set to much higher than media bit-rate. The server-to-intermediary and the intermediary-to-client sub-paths had average loss rates of 0.05 and 0.10, respectively, and with average burst length of 3. PSNR for `foreman` and `sean` sequences are given in Fig. 9 and 10, respectively. We see that the PSNR for *Optimized-SP* and *Simple* were comparable for larger display deadlines, but *Optimized-SP* significantly outperformed *Simple* at smaller display deadlines. Specifically, at display deadline of $200ms$, which allowed only one single retransmission in either but not both sub-paths, the PSNR gain was $2.7dB$ and $2.3dB$ for `foreman` and `sean`, respectively. The likely cause for the large PSNR gain at low delay was the insufficient time for retransmission which led to error propagation for *Simple*. On the other hand, as discussed in Section III, *Optimized-SP* could utilize late packets for the purpose of constructing secondary SP frames, leading to fewer occurrences of error propagation.

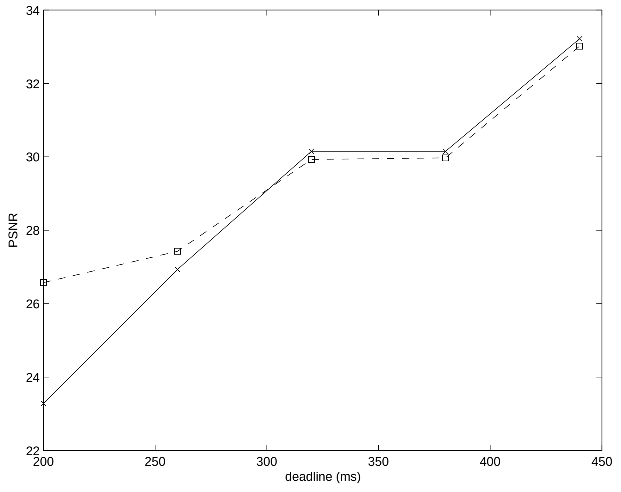

Fig. 9. PSNR of *Optimized-SP* (dashed) and *Simple* (solid) under different display deadlines for `foreman`.

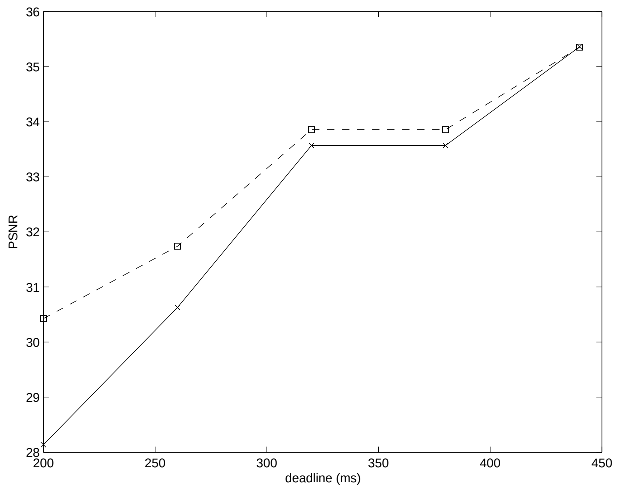

Fig. 10. PSNR of *Optimized-SP* (dashed) and *Simple* (solid) under different display deadlines for `sean`.

VI. CONCLUSION

In this paper, we presented a low-latency error control scheme that is applicable to implementation in a network intermediary. We showed associated formulation and optimization procedures of the scheme, and evaluated its performance using simulations. Our simulation results showed that significant PSNR improvement can be achieved by the scheme under tight delay constraints, or under medium delay constraints with small bandwidths. The paper demonstrated one practical way the new SP-frame feature of H.264 can be employed to improve video streaming in low-delay settings. Our scheme can be extended to the use of non-periodic SP-frames as well.

REFERENCES

[1] 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.

[2] L. Kontothanassis, R. Sitaraman, J. Wein, D. Hong, R. Kleinberg, B. Mancuso, D. Shaw, and D. Stodolsky, “A transport layer for live streaming in a content delivery network,” *Proceedings of IEEE*, September 2004.

[3] G. Cheung, W. t. Tan, and T. Yoshimura, “Double feedback streaming agent for real-time delivery of media over 3G wireless networks,” in *IEEE Transactions on Multimedia*, April 2004, vol. 6, no.2, pp. 304–314.

[4] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” in *IEEE Transactions on Circuits and Systems for Video Technology*, July 2003, vol. 13, no.7, pp. 560–576.

[5] J. Apostolopoulos, “Reliable video communication over lossy packet networks using multiple state encoding and path diversity,” in *Visual Communications and Image Processing*, January 2001.

[6] W. t. Tan and G. Cheung, “Using SP-frames for error resilience in optimized video streaming,” in *IEEE International Conference on Image Processing*, Atlanta, GA, October 2006.

[7] S. Floyd, M. Handley, J. Padhye, and J. Widmer, “Equation-based congestion control for unicast applications,” in *ACM SIGCOMM*, Stockholm, Sweden, August 2000.

[8] J. Castro, *The UMTS Network and Radio Access Technology*, John Wiley & Sons, 2001.