SP-Frame Selection for Video Streaming over Burst-loss Networks

Wai-tian Tan, Gene Cheung
Mobile & Media Systems Lab, Hewlett-Packard Laboratories

Abstract

SP-frame is a new picture type supported by H.264. The traditional usage of SP-frames is for switching between different compressed bit-streams. In this paper, we proposed and evaluated a scheme that uses SP frames as a mechanism to switch within a single compressed stream for the purpose of achieving error resilience and rate scalability. We have only considered the restricted but practical case in which only one secondary SP frame is allowed for every primary SP frame. Nevertheless, simulation results show that the technique can significantly increase the chance of video frames meeting their deadlines, and also improve overall PSNR.

1. Introduction

Many advances in video compression have been made to address the artifacts caused by data losses in the transport medium. These advances can be roughly categorized into two classes. The first class attempts to address error propagation caused by temporal prediction, and include techniques such as the use of periodic intra-frames and block, multiple description coding, and NewPred in MPEG-4. The second class attempts to minimize the amount of received data that is rendered useless by data loss. Examples include "video packet" (resync markers) and reversible VLC code in MPEG-4 error resilience tools. Error concealment is a common technique that sits between these two classes. In this paper, we investigate a novel approach in which S-frames are used to achieve two objectives, namely enhancing error-resilience by limiting error propagation, and providing rate-scalability for adaptation to time-varying channels.

Beyond the traditional Intra-frame (I-frame) and Inter-frame (P-frame) coding, a new frame type *S-frame* was introduced in the new video coding standard H.264 [5, 7]. There are two types of S-frames, namely the SI frame which is intra-frame coded, and the SP frame which employs temporal prediction. A brief survey of the encoding procedures is outlined in the next section, and interested readers are encouraged to read the references [5, 7]. A key characteristic of an SP frame, P, is that its reconstructed picture can be perfectly reproduced by another SP frame, S, even though S might predict from a different frame than P. For this reason, SP frames have been proposed for usage in applications involving switching between compressed bit-streams at boundaries not marked by intra-frames. An example is shown in Figure 1, where I_0 to P_6 and i_o to p_6 are two compressed bit-streams produced from the same content but coded at different bit-rates, with the sequence i_o to p_6 having lower bit-rate than I_0 to P_6. When the application determines that the transport medium can support a higher transport rate, a switch such as $i_o - p_1 - p_2 - p_3 - SP_s - P_5 - P_6$ that does not involve an intra-frame is possible. And Since SP_s perfectly reproduces the reconstructed picture of SP_4, it makes no difference whether P_5 is predicted from SP_4 or SP_s. The SP-frames SP_4 and sp_4 are often referred to as *primary* or *non-switching* SP-frames, while SP_s is referred to as a *secondary* or *switching* SP-frame.

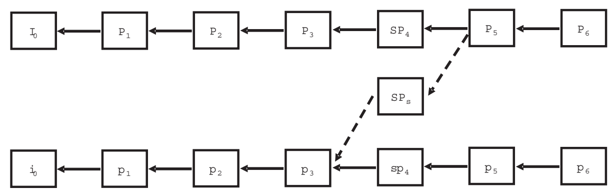


Figure 1. Using SP-frames to achieve switching between streams of the same video sequence coded at different bit-rates.

Clearly, *S-frames* when applied to different video sequences can effect switching at boundaries not involving an intra-frame. We have already seen in Figure 1 an example application of *S-frames* for switching between different bit-streams of the same video at different bit-rates. Nevertheless, *S-frames* can also be applied to a single bit-stream to effect switching *within* the bit-stream. An example is shown in Figure 2, where a 7-frame video sequence is encoded as $I_0 - P_1 - P_2 - P_3 - SP_4 - P_5 - P_6$. Note that frame 4 is encoded as a SP-frame, whose reconstructed picture can be perfectly reproduced by other *S-frames* such as SI, SP' and SP''. The frame SI is intra-coded, and SP' and SP'' have as reference frames P_2 and P_1, respectively. An important consequence is that for the reconstruction of P_5, and therefore subsequent pictures, it makes no difference whether P_5 is predicted from SP_4, SI, SP' or SP''. For instance, if I_0 and P_1 is received but P_2 through SP_4 are lost, the transmitter has the option of retransmitting all the lost frames and then P_5, or it can transmit SP'' or SI follow by P_5.

Clearly, persistent retransmission ensures that all frames are perfectly reconstructed at the receiver at the expense of having no ability to adapt to application needs. When the byte size of a secondary or switching SP frame is smaller than the byte size of the lost packets, it may sometimes be preferable to transmit the smaller SP frame at the expense that not all frames are perfectly reconstructed. In the context of Figure 2 where P_2 through SP_4 are lost, if the size of SP'' is smaller than the sum size of P_2 through SP_4, then sending SP'' is cheaper in terms of bytes to stop error propagation at P_5, but intermediate frames P_2 and P_3 cannot be reproduced perfectly. In this paper, we study the use of *S-frames* for switching within a single stream and evaluate the benefits of such schemes under different conditions.

The rest of the paper is organized as follows. Section 2 provides an overview of how *S-frames* achieve the important property to facilitate switching without intra-frames. Section 3 identifies some application scenarios in which the use of SP-frames provides a competitive solution. The formulation and procedures for the optimization scheme used for evaluation purposes in this paper are presented in Section 4. The results are then presented in Section 5 followed by a summary.

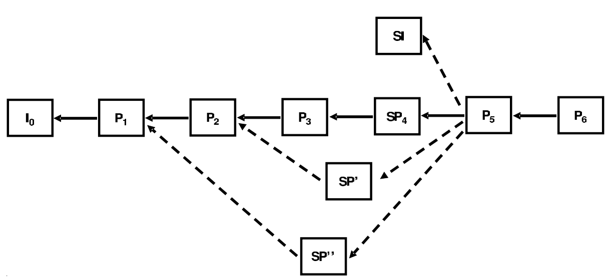


Figure 2. Usage Example of SP-Frames for Error Resiliency. The original sequence is transmitted as $I_0 - P_1 - P_2 - P_3 - SP_4 - P_5 - P_6$. The sequence $I_0 - P_1 - SP'' - P_5 - P_6$ allows perfect reconstruction of P_5 and effectively stopping error propagation due to loss of P_2, P_3 or SP_4.

2. Encoding of SP-Frames

In this section, we present a high level overview of the encoding process of primary and secondary SP frames to illustrate how they can achieve identical reconstructed picture despite using different reference frames.

A version of the primary SP frame encoder is described in [7] and illustrated abstractly in Figure 3. We denote frame F_i of type P using reference frame j as $P_j(i)$. It is essentially a typical Inter-frame encoder, with an added second quantization / de-quantization steps (Q_{SP} and Q_{SP}^{-1} in the figure) that quantized / de-quantized the primary SP frame ($S_{t-1}(t)$ in the figure) before the frame buffer. At the primary SP frame decoder, the reconstructed frame $S_{t-1}(t)$ is similarly quantized / de-quantized using Q_{SP} before being used for motion estimation and compensation for future frame $P_t(t+1)$. The quantized version of frame $S_{t-1}(t)$, $Q_{SP}(S_{t-1}(t))$, is the frame the secondary SP frame encoder needs to reproduce exactly for perfect reconstruction.

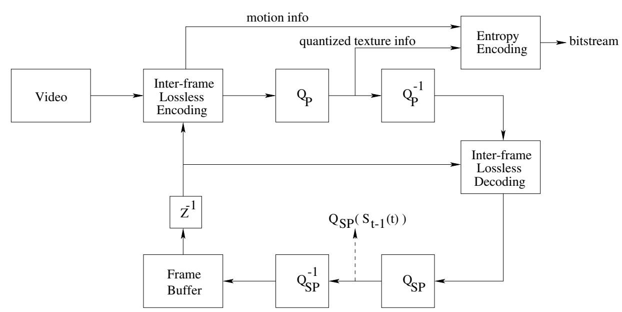


Figure 3. Primary SP Frame Encoder

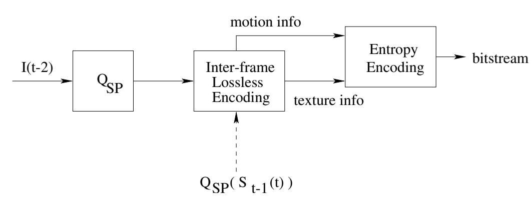


Figure 4. Secondary SP Frame Encoder

The goal of the secondary SP frame encoder then is to encode secondary SP frame — $S'_{t-2}(t)$ for example, using reference frame $I(t-2)$ so that *perfect* reconstruction of the quantized primary SP frame $S_{t-1}(t)$, $Q_{SP}(S_{t-1}(t))$, is possible at the decoder. To that end, reference frame $I(t-2)$ is first quantized using Q_{SP} before performing Inter-frame lossless encoding with quantized $S_{t-1}(t)$, or $Q_{SP}(S_{t-1}(t))$. This is shown in Figure 4. Notice there is no further quantization from this point, meaning that the decoder, with $I(t-2)$, Q_{SP} and the losslessly coded difference available, can perfectly recover $Q_{SP}(S_{t-1}(t))$.

Given the described pair of primary and secondary SP-frame encoders, it is clear that perfect reconstruction of the primary SP-frame is guaranteed regardless of the actual value of Q_{SP}. In practice, a coarser Q_{SP} would mean a poorer rate-distortion performance for the original encoded video sequence using primary SP-frames, while a finer Q_{SP} would necessitate a larger secondary SP-frame size.

3. Application Context

While there are many video streaming applications each with its own unique characteristics and requirements, in this paper we classify along two important dimensions, namely real-time versus non-real-time communication, and point-to-point versus point-to-multi-point communications. Clearly, both realtime communication and multi-point communication are additional requirements that characterize more challenging environments. In particular, realtime multi-receiver video communication, such as remote learning where a teacher maintains real-time communications with all students who have a real-time audio channel to ask questions, is the most challenging. In this section, we discuss two application contexts in which the use of *S-frames* provides signifcant advantage.

In non-real-time streaming environments with back channel, e.g., streaming playback of stored content, retransmission of loss data is arguably the most effective solution to counter data losses. There are two reasons. First, retransmission is efficient in that every packet is sucessfully transmitted only once. Second, an initial buffering delay for several seconds is acceptable, which is sufficient for many retransmission attempts to effectively render the channel apparently lossless. One drawback of persistent retransmission coupled with congestion control, as implemented in the ubiquitous transmission control protocol (TCP), is that the application loses control of the delivery time. Nevertheless, current generation of commercial streaming systems from Microsoft and Real Networks encode a piece of content into multiple bit-streams of different bit-rates and support coarse-grain switching between those to help ensure media data are delivered in time. This is one application context that *S-frames* are naturally useful, due to its ability to effect switching without I-frames. Even though an SP frame is larger in size than a P frame at the same quality, the percentage increase in bits due to the use of primary SP frames is generally negligible given a P-frame to SP-frame ratio of 10 and above [8].

For real-time applications such as video conferencing, the number of retransmission attempts for every piece of data is limited. To avoid error propagation, two common approaches are to use forward error correction (FEC) and increase the number of intra-coded frames or macro-blocks. The major drawback with FEC is the well-known fact that it is ineffective in channels with burst losses. The increased use of intra-coding, on the other hand, significantly increases the number of bits that needs to be transmitted. An effective solution for real-time point-to-point communication is to employ the method known as NewPred in MPEG-4 under which a client will notify the sender of lost data, and the sender will adjust its live encoder so that future frames will be produced in a way that is not dependent of the lost data. Unlike retransmission, there will be observable errors in the video stream under NewPred. Nevertheless, such errors will be short-lived.

Real-time communications to multiple recipients, such as in group video conference or the aforementioned remote learning application, proved to be challenging even for NewPred. This is because the model that a single live-encoder tailoring a customed stream for a single receiver falls apart. Specifically, a single sender now has to consider the aggregate loss of all receivers, which results in a bit-stream that is far from ideal for every receiver. The increased use of intra-coded frames or macroblocks is one solution but incurs an undue and high transmission cost for receivers with little or no losses. The use of intra-coded frames or macroblocks, on the other hand, is bandwidth efficient, but suffers from two problems. First is its poor ability to address error propagation due to data loss. Second is its ineffectiveness to support joining of new receivers as often happens in point-to-multi-point communication. The use of *S-frames* provides a practical compromise. An example is given in Figure 5 where a single sender is sending a sequence of video frames $I-P_0-P_1-P_2-P_3-SP_4-P_5$ to all receivers, possibly using multicast. Receiver A receives all transmitted packets and needs no further attention from the sender. Receiver A pays a small penalty due to the fact that an SP frame at the same quality is slightly larger than a P frame. Receiver B, on the other hand, suffer a number of losses. The sender in this case, can send a remedial frame SP_{05} to B to allow correct decoding of P_5. Note that this secondary SP frame SP_{05} is not sent to other receivers. If the size of SP_{05} is smaller than the sum size of the lost packets, then Receiver B saves on transmission cost. Finally, Receiver C joins the session after P_2 is delivered, and only sees packets $P_3-SP_4-P_5$. In this case the sender can transmit an SI frame to Receiver C to kick start decoding.

4. Optimized Streaming with S-frames

In this section, we discuss the settings under which we compare the optimal use of *S-frame* for switching within a single stream with purely retransmission-based schemes. We will first present the network and source models assumed, and then present a two-step optimization procedure.

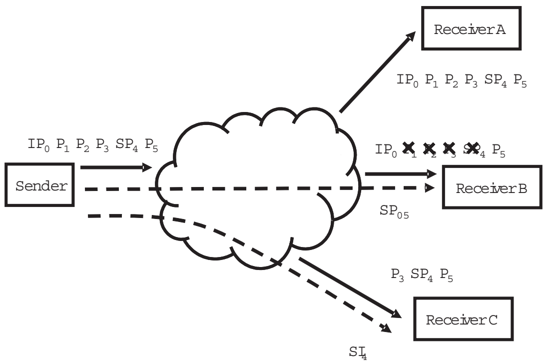


Figure 5. The inclusion of S-frames incurs a small bit cost for all streams but provide the flexibility for saving bandwidth when burst loss occurs, and the ability to quickly start decoding for receivers that join mid-session.

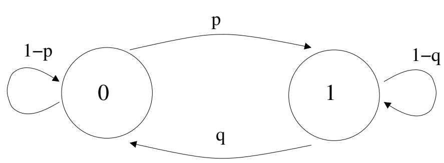


Figure 6. Gilbert Model for Packet Losses. By changing parameters p and q, different average packet loss rate and burst length can be achieved.

4.1. Network Model

For evaluation purpose, we assume a burst-loss network with loss process according to the Gilbert (two-state Markov) model as shown in Figure 6. We denote the packet delivery and packet loss events by 0 and 1, respectively. The parameters p and q are the respective state transition probabilities from the *delivery* and *loss* states. The average packet loss ratio (PLR) is given by $\pi = p/(p+q)$, and the average burst length is $1/q$.

Under the Gilbert loss model, we can derive expressions of some useful quantities as follows [4, 3]. We denote by $p(i)$, $i \geq 0$, the probability of having *exactly* i consecutive delivered packets between two lost packets, i.e. $p(i) = \Pr(0^i 1|1)$. We denote by $P(i)$ the probability of having *at least* i consecutive delivered packets following a lost packet, i.e., $P(i) = \Pr(0^i|1)$. Then, $p(i)$ and $P(i)$ can be written mathematically as:

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

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

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

$$Q(i) = \begin{cases} 1 & \text{if } i=0 \\ p(1-q)^{i-1} & \text{o.w.} \end{cases} \quad (4)$$

where the complementary functions $q(i)$ and $Q(i)$ are defined as follows: $q(i) = \Pr(1^i 0|0)$ 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:

$$R(m,n) = \begin{cases} P(n) & \text{for } m=0 \text{ and } n \geq 0 \\ \sum_{i=0}^{n-m} p(i)R(m-1, n-i-1) & \text{for } 1 \leq m \leq n \end{cases} \quad (5)$$

We additionally define $r(m,n)$ as the probability that there are *exactly* m lost packets in n packets *between* two lost packets following an observed lost packet. Similarly, $r(m,n)$ can be expressed recursively:

$$r(m,n) = \begin{cases} p(n) & \text{for } m=0 \text{ and } n \geq 0 \\ \sum_{i=0}^{n-m} p(i)r(m-1, n-i-1) & \text{for } 1 \leq m \leq n \end{cases} \quad (6)$$

Finally, we define $\bar{r}(m,n)$ as the probability that there are *exactly* m lost packets in n packets following a lost packet and preceding a successfully received packet:

$$\bar{r}(m,n) = R(m,n) - r(m,n) \quad (7)$$

We define the complementary function $S(m,n)$ as the probability of having *exactly* m delivered packet out of n transmitted packets following a delivered packet. We have under the Gilbert model:

$$S(m,n) = \begin{cases} Q(n) & \text{for } m=0 \text{ and } n \geq 0 \\ \sum_{i=0}^{n-m} q(i)S(m-1, n-i-1) & \text{for } 1 \leq m \leq n \end{cases} \quad (8)$$

$s(m,n)$ and $\bar{s}(m,n)$ are defined as counterparts to $r(m,n)$ and $\bar{r}(m,n)$.

For network transport, we assume a transmission bandwidth of C kbps and a fixed maximum transmission unit of MTU bytes per packet.

For network delay, we assume a shifted-Gamma-distributed random variable delay $\gamma \sim \mathcal{G}(\kappa, \alpha, \lambda)$ with probability density function (pdf):

$$g_{\Gamma_s}(\gamma) = \frac{\lambda^{\alpha}(\gamma-\kappa)^{\alpha-1} e^{-\lambda(\gamma-\kappa)}}{\Gamma(\alpha)} \qquad \kappa < \gamma < \infty \quad (9)$$

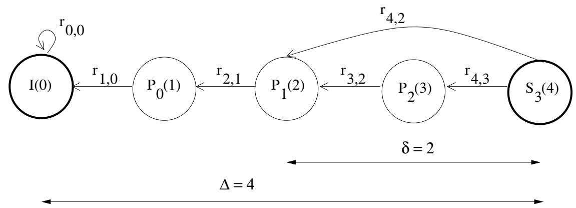


Figure 7. DAG Source Model

where $\Gamma(\alpha)$ is the *Gamma function*:

$$\Gamma(\alpha) = \int_0^\infty \tau^{\alpha-1} e^{-\tau} d\tau \qquad \alpha > 0 \tag{10}$$

4.2. Source Model

Similar to our earlier work on reference frame selection [1, 2], we assume each frame F_i is represented by a node in an acyclic directed graph (DAG) as shown in Figure 7. Associated with each node i is a set of edges $E_{i,j}$'s that point to the reference frames F_j's that F_i can use for motion compensation. For a P-frame, there is only one edge, while for SP-frames, there are two reference frames corresponding to primary and secondary SP-frames, respectively. Associated with each node i is a delivery deadline T_i, upon which the frame F_i must be delivered to the client or it will be rendered useless. Associated with each edge $E_{i,j}$ is a rate term $r_{i,j}$ specifying the number of bytes the encoder requires to encode F_i using F_j as reference. As expected, the larger the temporal distance between F_i and F_j, the larger $r_{i,j}$ likely will be.

In addition, we assume the following structure for SP-frames. First, an SP-frame is inserted into the video sequence every Δ_{SP} frames. Second, a secondary SP-frame i uses frame $i - \delta_{SP}$ as reference when performing motion prediction and compensation. Figure 7 shows an example when $\Delta_{SP} = 4$ and $\delta_{SP} = 2$. Δ_{SP} and δ_{SP} are parameters we will optimize during pre-encoding of the video.

Finally, we assume the video sequence has a playback speed of FPS frames per second and an initial client buffering delay of BUF seconds.

4.3. Problem Definition

Given the abstract models we have discussed in previous sections, we are interested in the following two problems. First, given network condition, how to select Δ_{SP} and δ_{SP} for SP-frames *a priori* such that appropriate flexibility is available during the streaming session to counteract potential packet loss. Second, during real-time streaming session, how to select the correct frames for (re)transmission during each transmission opportunity, given observed network conditions and client feedbacks. We address these two problems separately in the next two subsections.

4.4. Optimized offline Encoding

For simplicity, we first assume primary SP-frame $S_{i-1}(i)$ is encoded with the same quantization parameter as P-frame $P_{i-1}(i)$. We next assume a storage space limit V^* in bytes for the pre-encoded video. Assuming there are N inter-coded frames following an I-frame, the storage constraint is expressed as:

$$r_{0,0} + \sum_{i=1}^{N} r_{i,i-1} + \sum_{k=1}^{\left\lfloor \frac{N}{\Delta_{SP}} \right\rfloor} r_{k\Delta_{SP},\, k\Delta_{SP}-\delta_{SP}} \leq V^* \tag{11}$$

where the three terms on the left-hand side are the sizes of the I-frame, the N P-frames and the $\left\lfloor \frac{N}{\Delta_{SP}} \right\rfloor$ secondary SP-frames, respectively.

Intuitively, Δ_{SP} negatively influences storage size: large Δ_{SP} means fewer SP-frames. In contrast, δ_{SP} positively influences storage size: large δ_{SP} means larger temporal distance between secondary SP-frame F_i and reference frame $F_{i-\delta_{SP}}$, and hence more bits are needed.

Our goal for the off-line optimization as follows: given storage constraint (11), find Δ_{SP} and δ_{SP} that maximize the probability that a given video sequence is *synchronized*, meaning that all transmitted frames are timely delivered and correctly decoded. To maximize this quantity for given Δ_{SP} and δ_{SP}, we assume the server sends only *essential* frames — frames that future frames depend on. In Figure 7, the essential frames are F_0, F_1, F_2 and F_4. We choose this metric for the off-line optimization because a synchronized video sequence containing only essential frames represents the minimum set of frames for the video to be continuously decodable.

For each Δ_{SP} and δ_{SP}, we calculate the probability that the sequence is synchronized up to essential frame F_i as follows. We first compute the number of packets h_i that needs to be transmitted for frame F_i, where:

$$h_i = \begin{cases} \lceil r_{i,i}/MTU \rceil & \text{for I-frame} \\ \lceil r_{i,i-1}/MTU \rceil & \text{for P frames} \\ \lceil r_{i,i-\delta_{SP}}/MTU \rceil & \text{for secondary-SP frame} \end{cases} \tag{12}$$

We then compute the average packet size, s_{pkt} as:

$$s_{pkt} = \frac{r_{0,0} + \sum_{i=1}^{N} r_{i,i-1}\; I(i) + \sum_{k=1}^{\left\lfloor \frac{N}{\Delta_{SP}} \right\rfloor} r_{k\Delta_{SP},\, k\Delta_{SP}-\delta_{SP}}}{\sum_{i=0}^{N} h_i\; I(i)} \tag{13}$$

where $I(i)$ is an indicator function that equals 1 when F_i is an essential frame, and 0 otherwise. The first frame, with h_0

packets, has to be delivered within the first BUF seconds, corresponding to k_0 transmission opportunities:

$$k_0 = \lfloor BUF * (1000 * C/8)/s_{pkt} \rfloor \tag{14}$$

Given the Gilbert network loss model, the probability that packets of F_0 are delivered on time, L_0, is then:

$$L_0 = \pi \sum_{i=0}^{k_0-h_0} R(i,k_0) + (1-\pi) \sum_{i=h_0}^{k_0} S(i,k_0) \tag{15}$$

In turn, F_1 will have $1/FPS$ seconds worth of transmission opportunities *plus* leftover opportunities k_1 from initial buffering not already spent for delivery of F_0. In general, $k_i, i \geq 1$, is a random variable whose probability mass function (pmf) we denote by $P_i(k)$. Define the starting $P_0(k)$ to be equal to 1 if $k = k_0$, and 0 otherwise. Suppose that the number of leftover transmission opportunities from previous frame time k_{i-1} is j. Then $k_i = k$ if *exactly* j *plus* t seconds worth of transmission opportunities (g) *minus* k were spent to correctly deliver h_{i-1} packets to the client; t is $1/FPS$ if F_{i-1} is a P frame, and δ_{SP}/FPS if F_{i-1} is a secondary SP-frame. Note that this assumes the last packet transmission attempt of F_{i-1} is always successful. Hence we will only consider the prior $h_{i-1}-1$ correct packet deliveries in one fewer total packet delivery attempts. Mathematically, we write:

$$g = \begin{cases} 1/FPS * (1000 * C/8)/s_{pkt} & \text{for P frame} \\ \frac{\delta_{SP}}{FPS} * (1000 * C/8)/s_{pkt} & \text{for secondary SP-frame} \end{cases}$$

$$\begin{aligned} P_i(k) = \frac{1}{\bar{P}_{i-1}} \sum_{j=0}^{k_0} & P_{i-1}(j)\, \pi\, \bar{r}(g+j-k-h_{i-1},\ g+j-k-1) \\ & + P_{i-1}(j)\,(1-\pi)\, s(h_{i-1}-1,\ g+j-k-1) \end{aligned}$$

where $\bar{P}_{i-1} = \sum_{j=0}^{k_0} P_{i-1}(j)$ is needed for normalization. Given $P_i(k)$, we can now write L_i as:

$$L_i = \sum_{k_i=0}^{k_0} P_i(k_i) \left(\pi \sum_{j=0}^{k_i+g-h_i} R(j, k_i+g) + (1-\pi) \sum_{j=h_i}^{k_i+g} S(j, k_i+g) \right) \tag{16}$$

We seek to maximize the product of all L_i of essential P-frames and secondary SP-frames:

$$\max_{\Delta_{SP}, \delta_{SP}} \prod_{i=1}^{N} L_i^{I(i)} \tag{17}$$

Given this is an offline optimization, our approach is to exhaustively search all reasonable Δ_{SP} and δ_{SP} to find the parameters that maximize (17) while satisfying (11).

4.5. Optimized Real-time Streaming

Given pre-encoding of the video sequence using selected Δ_{SP} and δ_{SP}, the objective in this section is to determine which packet of which video frame to transmit for each transmission opportunity. By transmission opportunity, we mean that given average packet size of s_{pkt} and bandwidth C kbps, we can send a packet at a minimum time interval of every T_{avg} seconds:

$$T_{avg} = \frac{s_{pkt}}{1000 * C/8} \tag{18}$$

Hence the streaming server has a transmission opportunity every T_{avg} seconds. We assume that the client sends a NACK message to the streaming server upon detection of a packet loss. We further assume that those NACK messages are not lost. Without receiving any NACKs from the client, the server will stream the video frame-by-frame in sequence. When a NACK is received by the server indicating a packet j of frame F_l is lost, the server has to make one of two choices: i) retransmit packet j of frame F_l, ii) skip ahead and send the next secondary SP-frame instead. The first choice attempts to completely reconstruct the video sequence, while the second choice forgoes frame F_l up to the P-frame before the next secondary SP-frame, to ensure transmission of the SP-frame and hence maximizing synchronization probability. If the reported lost packet is of frame F_l that is before or the same as the reference frame the secondary SP-frame is using, then obviously the server needs to retransmit it to ensure synchronization. In the example of Figure 7, if a packet from frame $P_0(1)$ or frame $P_1(2)$ is missing, retransmission is required. The problem is when the lost packets are of P-frames after the P-frame that SP-frame is referencing, what decision should the server make to optimize streaming performance.

First, let a be the total number of packets from F_l up till and including the next primary SP-frame. If we let $x = \left(\left\lfloor \frac{l}{\Delta_{SP}} \right\rfloor + 1 \right) \Delta_{SP}$ be the index of the next SP-frame, we can write:

$$a = \sum_{i=l}^{x} \left\lceil \frac{r_{i,i-1}}{MTU} \right\rceil \tag{19}$$

In addition, let b be the number of packets for the next secondary SP-frame, written as:

$$b = \left\lceil \frac{r_{x,x-\delta}}{MTU} \right\rceil \tag{20}$$

As done previously, we estimate the number of packets K that the server can send until frame F_x is expected at the client. We assume that the reception of an NACK packet indicates that we are currently in the bad network state of the Gilbert model. The expected number of correctly decoded frames H_{ARQ} by retransmitting F_l till F_x, for frames F_l up till F_x is given by:

$$H_{ARQ} = (x-l+1) \sum_{i=0}^{K-a} R(i,K) \tag{21}$$

Table 1. Network Parameters for Experiment

p	0.037037 (trial 1)	0.02222 (trial 2)
q	0.3333 (trial 1)	0.2 (trial 2)
C	variable	
MTU	1500 bytes	
κ	50 ms	
α	4	
λ	0.2	

Similarly, we can calculate the expected number of decoded frames H_{SP} if we use the next secondary SP-frame to skip ahead, for the single frame F_x:

$$H_{SP} = \sum_{i=0}^{K-b} R(i, K) \tag{22}$$

The decision rule we employ is simply the following: if H_{ARQ} is larger than H_{SP}, we retransmit; otherwise, we skip to the next SP-frame.

5. Results

To examine the effectiveness of the proposed optimizations, we constructed a network simulator written in `C`. The network model parameters used in the experiments are shown in Table 1. The Gilbert parameters p and q are selected so that the PLR is 0.1 for both trials, while the burst lengths are 3 and 5, respectively.

For sources, we used two standard MPEG video test sequences in QCIF, `foreman` and `sean`, at 10 fps and with quantization parameters 26, 24 and 25 for P-frames, primary SP-frames and secondary SP-frames, respectively. We use JM 7.6 for encoding primary SP frames, and public domain software by Eric Setton of Stanford University to generate secondary SP frames [6].

We first look at results for the offline optimization. Figures 8 and 9 shows the results for `foreman` at an expected loss rate of 10% and average burst lengths of 3 and 5, respectively. As expected, the storage cost increases with δ. This is expected because the size of the secondary SP frame increases with δ. Also, storage cost decreases with increasing Δ. This is due to two reasons. First, there is a small inefficiency in using SP frames compared to P frames. A large Δ values causes the percentage of frames coded as SP to decrease. Second, more primary SP frames also means that more secondary SP frames need to be generated and stored, prompting drastic increase in storage cost. We also observe that synchronization success probability is generally low when δ is small compared to Δ. This can be explained by the fact that a small δ provides little bandwidth savings by sending the secondary SP frame. In the context of Figure 7 with δ=2, there is little savings by sending secondary SP frame of rate $r_{4,2}$ than frames P_2 and S_3 with

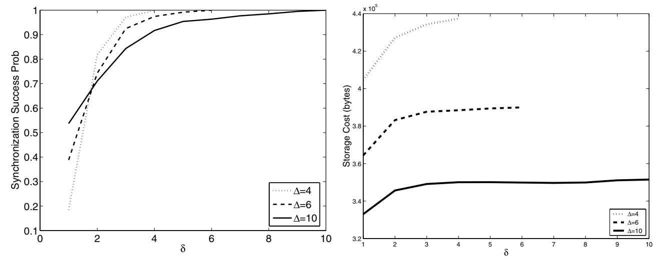


Figure 8. Performance Comparisons for `foreman` sequence at 10% loss and burst length of 3.

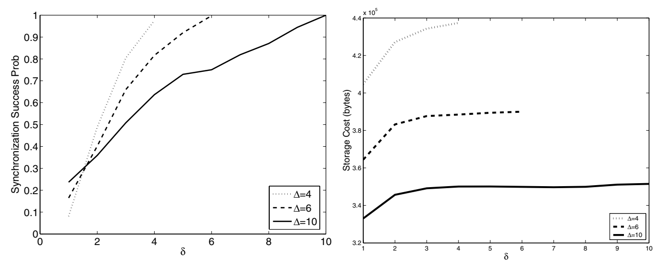


Figure 9. Performance Comparisons for `foreman` sequence at 10% loss and burst length of 5.

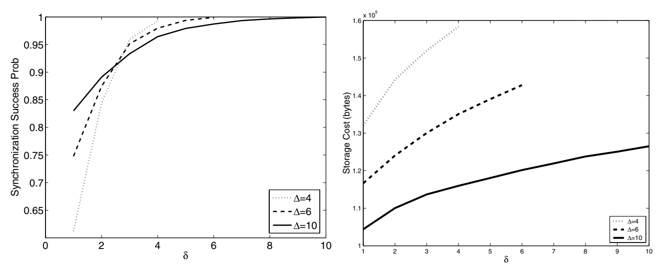


Figure 10. Performance Comparisons for `sean` sequence at 10% loss and burst length of 3.

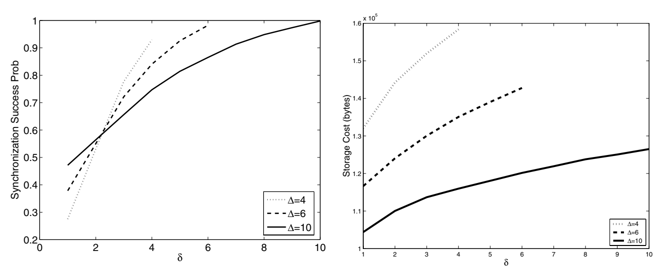
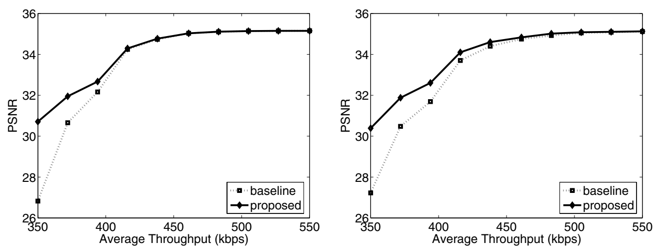


Figure 11. Performance Comparisons for `sean` sequence at 10% loss and burst length of 5.

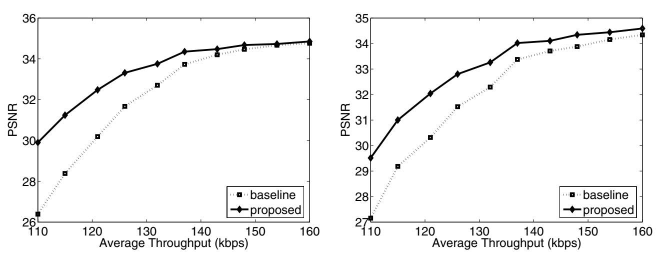
combined rate of $r_{3,2} + r_{4,3}$. Generally, the highest resynchronization probability is obtained when Δ and δ are comparable. This is expected since that corresponds to the case when the least amount of data needs to be transmitted to retain resynchronization. Corresponding results for the `Sean` sequence are shown in Figures 10 and 11.

We next evaluate the performance of using SP frames for online adaptive streaming. For the purpose of comparison, a `baseline` retransmission scheme is constructed where I-frame, P-frames and primary SP-frames are transmitted in order, and any NACKs will trigger retransmission until successful delivery.



a) Burst-length = 3 b) Burst-length = 5

Figure 12. Performance Comparisons for `foreman` sequence



a) Burst-length = 3 b) Burst-length = 5

Figure 13. Performance Comparisons `sean` Sequence

The performance for the two schemes for the `foreman` sequence is shown in Figure 12. The performance is shown in PSNR as function of the available bandwidth. We see that our proposed scheme `proposed` outperformed `baseline` for all ranges of the bandwidth for both burst lengths . We see that performance difference is significant when available bandwidth is low, but negligible with plentiful bandwidth. This is expected since both `baseline` and `proposed` assume the same strategy of not transmitting secondary SP frames with plentiful bandwidth.

The performances for the two schemes for the `sean` sequence is shown in Figure 13, for both trials. Similarly, we see that our proposed scheme `proposed` outperformed `baseline` for all ranges of the bandwidth for both burst lengths .

6. Summary

Traditionally, SP frames are used for switching between different compressed bit-streams. In this paper, we proposed and evaluated a scheme use SP frames as a mechanism to switch *within* a single compressed stream for the purpose of achieving better error resilience. We have only considered the restricted but practical case in which only one secondary SP frame is allowed for every primary SP frame. Nevertheless, results show that significant PSNR improvement can be achieved over a wide range of operating conditions.

References

[1] G. Cheung and W. t. Tan. Reference frame optimization for multi-path video streaming using complexity scaling. In *International Packet Video Workshop*, Irvine, CA, December 2004.

[2] G. Cheung and W. t. Tan. Loss-compensated reference frame optimization for multi-path video streaming. In *IEEE International Conference on Multimedia and Expo*, Amsterdam, the Netherlands, July 2005.

[3] P. Frossard. FEC performance in multimedia streaming. In *IEEE Communications Letters*, volume 5, no.3, March 2001.

[4] P. Frossard and O. Verscheure. Joint source/FEC rate selection for quality-optimal MPEG-2 video delivery. In *IEEE Trans. Image Processing*, volume 10, no.12, pages 1815–1825, December 2001.

[5] 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*, volume 13, no.7, July 2003.

[6] E. Setton. `http://www.stanford.edu/~esetton/H264.htm`.

[7] X. Sun, S. Li, F. Wu, J. Shen, and W. Gao. The improved SP frame coding technique for the JVT standard. In *IEEE International Conference on Image Processing*, Barcelona, Spain, September 2003.

[8] X. Sun, F. Wu, S. Li, W. Gao, and Y. Zhang. Improved SP coding technique, February 2002. ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6.