# EXPECTED RUN-TIME DISTORTION BASED SCHEDULING FOR SCALABLE VIDEO TRANSMISSION WITH HYBRID FEC/ARQ ERROR CONTROL

*Tong Gan*[1], *Lu Gan*[2] *and Kai-Kuang Ma*[3]
[1] Interuniversity Microelectronics Center, Leuven, Belgium
Email: gantong@imec.be
[2] School of Electrical Engineering and Electronics, The University of Liverpool, Liverpool, UK
Email: ganlu@ieee.org
[3] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
Email: ekkma@ntu.edu.sg

## ABSTRACT

The optimal packet scheduling for transmitting scalable media over the packet erasure networks has been extensively studied in the past. In the existing work, only retransmission is used for packet loss recovery. As a result, when the *round-trip-time* (RTT) of the network increases, the performance of the system degrades fast. In this paper, a scheduling scheme for hybrid FEC/ARQ error control is proposed. In our proposed scheme, the importance of both data packets and FEC packets are evaluated by considering several factors, such as data dependency structure of scalable video, transmission history of other packets, as well as the decoding deadline, and the most important packet is chosen to be sent out. In this way, our proposed scheme is able to achieve more stable playback video quality by taking advantage of both FEC and ARQ, as demonstrated by the experimental results.

***Index Terms*—** scalable video streaming, FEC, ARQ, hybrid error control, packet scheduling

## 1. INTRODUCTION

With the fast advancement of both communication and video compression technologies, video streaming is becoming more and more popular. Since the available bandwidth in the communication networks is often time-varying, it is thus desirable to employ scalable video coding, such as MPEG-4 *fine granular scalability* (FGS) [1], to provide rate adaptation. With scalable video coding, the source video is encoded into a base layer and multiple enhancement layers. Briefly, the base layer provides a coarse picture quality, while the enhancement layers can be decoded progressively to refine the video quality.

When transmitting scalable video across the networks, there are two main challenges. First, since the current networks only provide best-effort *quality-of-service* (QoS), packet losses are inevitable. Second, there is no differentiation of packet importance. Base layer packets (which are more important in video decoding) will be treated the same as enhancement layer packets.

In order to address these two challenges, many error control mechanisms have been proposed in the literature. First, in order to recover packet loss, *forward error correction* (FEC) and retransmission (such as *auto repeat request*, ARQ) are widely used. Generally, FEC is more suitable for applications with stringent end-to-end delay requirement, but the parity packets will introduce additional redundancy. On the other hand, ARQ only retransmits lost packets (hence no redundancy), but it may not be a viable solution if the *round trip time* (RTT) is relatively large compared to the start-up delay. In view of this, some hybrid FEC/ARQ error control schemes (e.g., [2]-[3]) have been developed to take advantage of both mechanisms, and these schemes generally have better performance than purely using FEC or ARQ.

Secondly, to handle the nonuniform importance structure of scalable video, Miao *et al.* [4] proposed a retransmission based scheme, called *expected run-time distortion based scheduling* (ERDBS). Since ERDBS only adopts ARQ, it becomes inefficient with larger RTT. Specifically, according to the reported results [4], when the RTT is increased from 100ms to 300ms, the PSNR of the playback video is decreased by $8dB$. Given that the RTT in the Internet is time-varying and could range from 100ms to 300ms or more, the playback video quality could be rather unstable using the existing scheme [4].

In this paper, we make an extension to the ERDBS scheme [4] by incorporating FEC into error control. A scheduling algorithm for the hybrid FEC/ARQ scheme is proposed, which is denoted by ERDBS-H. In our proposed scheme, the source data packets are first encoded offline using systematic error-correction code (e.g., the *Reed-Solomon* code) to generate parity packets. During transmission, the server has to evaluate the importance of each packet in the candidate set, including both new packets and lost packets. Note that these packets could be either data packets or parity packets. In this way, our proposed scheduling scheme is able to achieve a balance between FEC and ARQ.

The rest of this paper is organized as follows. In Section 2, a brief review of the ERDBS scheme is presented, which serves as the foundation of our work. In Section 3, our proposed ERDBS-H scheme is described in detail. Some preliminary experimental results are be shown in Section 4, followed by conclusions in Section 5.

## 2. THE BASICS OF ERDBS

When ARQ is adopted in a scalable video streaming system, there are two types of packets in the server's buffer that could be sent out, namely, new data packets that have never been sent out before and lost packets which need to be retransmitted. The fundamental problem is thus to find an optimal schedule for the delivery of packets so that the playback video quality at the client side can be maximized. Some algorithm has been proposed in [5] to obtain global optimal solutions, but the high computational complexity may prohibit its use in a practical communication system. In [4], a new metric called *expected run-time distortion* (ERD) is proposed to evaluate the importance of each packet. The most important packet is then chosen from the candidate set and transmitted. It has been shown that this

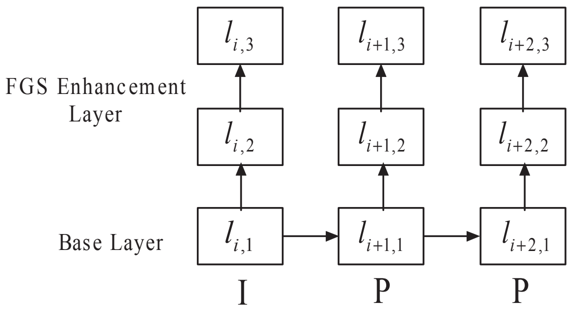


**Fig. 1**. Data dependencies among the IPPP frames. The arrow is directed from a parent packet to a child. (Adapted from [4])

low-complexity heuristic scheduling algorithm can achieve comparable performance to the globally optimal algorithm [5]. Therefore, it becomes an attractive approach for practical implementation.

There are two main concepts in ERDBS, i.e., scalable video data structure modelling and the ERD. In the following sections, the notations closely follow that in [4].

### 2.1. Scalable video data structure modeling

Without loss of generality, let's take MPEG-4 FGS encoded video as an example. For each frame, there is a base layer and a FGS enhancement layer which can be truncated arbitrarily to provide fine granular scalability. Suppose the bitstream of the $i^{th}$ frame is packetized such that the base layer is encapsulated into one packet, and the enhancement layer is divided into multiple packets. Let $l_{i,j}$ denote the $j^{th}$ packet of the $i^{th}$ frame. With such a packetization scheme, the data dependencies between different frames and layers can be modelled using a direct graph. As an example, Figure 1 illustrates the data dependencies among the IPPP frames, where the arrow is directed from a *parent* packet to a *child* packet. The philosophy behind this graph is that a child packet can only be decoded successfully if its parent packet is decodable. Bear this in mind, the *parent set* $A_{i,j}$ of $l_{i,j}$ is defined as the set of packets that need to be present in order to decode $l_{i,j}$, and the *child set* $B_{i,j}$ is the set of packets that can not be decoded without $l_{i,j}$. For instance, in Figure 1, the parent set of $l_{i+2,2}$ is $A_{i+2,2} = \{l_{i,1}, l_{i+1,1}, l_{i+2,1}\}$; while the child set of $l_{i+1,1}$ is $B_{i+1,1} = \{l_{i+1,2}, l_{i+1,3}, l_{i+2,1}, l_{i+2,2}, l_{i+2,3}\}$

### 2.2. Expected run-time distortion

The contribution of each packet in video decoding can be measured using the rate-distortion curve, as illustrated in Figure 2. Here, the size of $l_{i,j}$ is $r_{i,j}$ (bytes), and if it is decoded, the distortion of the $i^{th}$ frame can be decreased by $d_{i,j}$. However, this metric doesn't take into account the data dependency structure in Figure 1 and the transmission history (i.e., which packets were transmitted and whether they are received). To address this issue, Miao *et al.* proposed a new concept called run-time distortion, $\hat{d}_{i,j}$, which can be calculated as follows,

$$\hat{d}_{i,j} = d_{i,j} \prod_{l_{u,v} \in A_{i,j}} P\left(l_{u,v}\right) + \sum_{l_{u,v} \in B_{i,j}} d_{u,v} P\left(l_{u,v}\right) \quad (1)$$

where $P\left(l_{u,v}\right)$ is the probability that packet $l_{u,v}$ is able to be received by the client based on the transmission history. To be more specific, $P\left(l_{u,v}\right)$ can be obtained using (2), where $\epsilon$ is the mean packet-loss rate in the communication channel. In (1), the first term

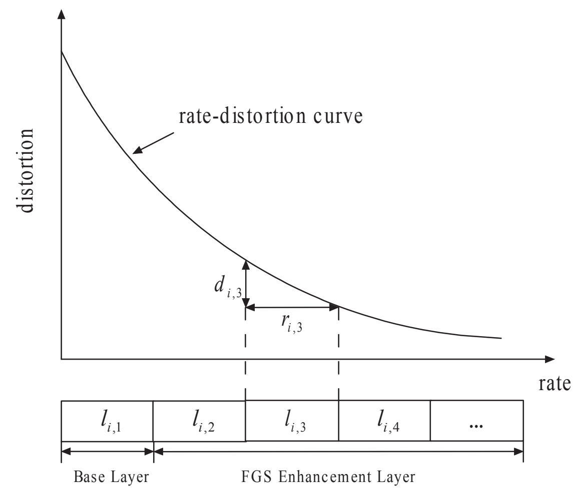


**Fig. 2**. Contribution of each packet in video decoding. After decoding $l_{i,3}$ which contains $r_{i,3}$ bytes, the distortion of the $i^{th}$ frame is reduced by $d_{i,3}$. (Adapted from [4])

implies that $l_{i,j}$ can only be decoded given that all its parents are present, and the second term indicates that the importance of $l_{i,j}$ becomes more significant if any of its child packets have been received by the client.

Further more, the importance of packet $l_{i,j}$ is also affected by its decoding deadline. let $\tau$ denote the start-up delay, which is the interval between the instance when the first packet arrives at the client and the instance when the first frame is decoded. The decoding deadline of the $i^{th}$ frame, $t_i$, is thus

$$t_i = \tau + i \cdot \Delta \quad (3)$$

where $\Delta$ is the display interval between two frames. Intuitively, the closer the deadline, the more important is the packet, as it will have less chance for being retransmitted. By taking this into consideration, the run-time distortion $\hat{d}_{i,j}$ is weighted by a factor $w_{i,j}$ to reflect its decoding urgency, and the resulting metric is defined as *expected run-time distortion* (ERD), denoted by $\tilde{d}_{i,j}$,

$$\tilde{d}_{i,j} = w_{i,j} \times \hat{d}_{i,j} \quad (4)$$

In [4], the urgency factor $w_{i,j}$ is calculated using

$$w_{i,j} = \epsilon^{\frac{t_i - t - \frac{r_{i,j}}{C}}{RTT}} \quad (5)$$

where $t$ is the current time, $C$ is the available bandwidth, and $RTT$ is the round-trip time. Note that $\epsilon$, $C$ and $RTT$ could all be time varying.

In ERDBS, the ERD is calculated for each candidate packet, and the most important one will be sent out.

## 3. ERDBS-H FOR HYBRID ERROR CONTROL SCHEME

The major drawback of ERDBS lies in the fact that it only adopts ARQ for packet loss recovery. Thus, when the RTT becomes larger, the performance of the system will be severely degraded. To deal with this problem, FEC is incorporated in our proposed scheme. To be more specific, the packets of each frame would be protected with $(N, k)$ RS codes. That is, for each $k$ data packets, $N - k$ parity

$$P\left(l_{u,v}\right)=\begin{cases}1, & \text{if } l_{u,v} \text{ is ACKed;}\\ 1-\epsilon^{n}, & \text{if } l_{u,v} \text{ has been sent out } n \text{ times without ACK;}\end{cases} \tag{2}$$

packets are generated, as is usually done in the literature. Hereafter, these $N$ packets are referred to as a *block of packets* (BOP). With RS coding, when *any* $k$ packets out of the total $N$ packets are received, the $k$ data packets can be recovered.

During transmission, the client will acknowledge the receipt of each packet by sending ACK to the server. After sent out a packet, if the server doesn't receive ACK after the time-out threshold, say, one RTT, the server would assume that the packet get lost and would put that packet into the sending buffer. Thus, upon each transmission slot, the server will scan the candidate packets in the sending buffer, including both new packets and retransmit packets, and find out the most important packet. To this end, the importance of each packet has to be evaluated.

### 3.1. Expected run-time distortion of data packet

The run-time distortion of each data packet can be obtained using (1). What differs from the original ERDBS is the computation of $P\left(l_{u,v}\right)$. Consider the BOP of which $l_{u,v}$ belongs to. Let $n^{ACK}$ denote the number of unique packets received by the client of this BOP. Also, let $n^{OUT}$ denote the number of unique outstanding packets (i.e., those packets being sent out without ACK) of this BOP. Here, unique means that if a packet was sent out (or ACKed) multiple times, they are only counted once. The probability that $l_{u,v}$ can be received (or recovered after channel decoding) by the client, $P\left(l_{u,v}\right)$, could be obtained using (6). In (6), $p\left(k-n^{ACK}, n^{OUT}-1\right)$ is the probability that equal or more than $k-n^{ACK}$ out of $n^{OUT}-1$ packets can be received, and $\epsilon^{n}\cdot p\left(k-n^{ACK}, n^{OUT}-1\right)$ counts for the probability that $l_{u,v}$ is lost but can be recovered using FEC. Assume that the channel packet loss is i.i.d. with a mean loss rate of $\epsilon$, $p\left(k-n^{ACK}, n^{OUT}-1\right)$ can be easily calculated using (7).

Furthermore, the run-time distortion of $l_{i,j}$ should be weighted by the urgency factor, $w_{i,j}$ to get the ERD as in (4). Here, $w_{i,j}$ is calculated as

$$w_{i,j}=\frac{1}{t_i-t-\frac{r_{i,j}}{C}} \tag{8}$$

which implies that the closer the decoding deadline, the more urgent is the delivery of the packet.

### 3.2. Expected run-time distortion of parity packet

The run-time distortion of parity packets can be acquired by considering the following three scenarios. Again, consider the BOP which the parity packet belongs to.

Scenario 1:

If equal or more than $k$ packets are received by the client for this BOP, the arrival of the parity packet can not make any contribution, as the number of packets received is already enough to recover the $k$ data packets.

Scenario 2:

If $k-1$ packets are received by the client for this BOP, the arrival of this parity packet would enable the client to recover all data packets that are not ACKed. In this case, the run-time distortion of this parity packet would be the sum of those data packets to be recovered.

Scenario 3:

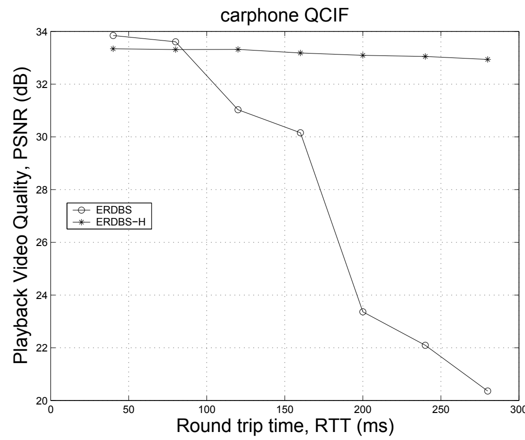


**Fig. 3**. Performance of the scheduling schemes under different values of RTT. Start-up delay is 200ms and the mean packet loss rate is 20%.

If less than $k-1$ packets are received, the arrival of this parity packet couldn't make any contribution to channel decoding. Thus it's also useless.

In summary, the run-time distortion of a parity packet, $\hat{d}_{FEC}$ can be obtained using (9), where $G_{NACK}$ is the set of data packets (of the same BOP as the parity packet) that are not ACKed.

Finally, $\hat{d}_{FEC}$ is weighted by the same urgency factor in (8) to obtain the ERD of the parity packet.

## 4. EXPERIMENTAL RESULTS

In this section, we present some experimental results for the proposed ERDBS-H scheduling scheme. The test sequence is the QCIF "Carphone" video, and is encoded using MPEG-4 FGS with IPPP... structure at 10 frames per second. The channel bandwidth is $C = 600kbps$ and the packet loss is assumed to be i.i.d.. Before transmission, the packets in each frame is channel coded using (10, 5) RS codes. The performance of the scheme is studied under different values of packet-loss rate, start-up delay, and RTT. For each set of parameters, the simulation is conducted for more than 100 times, and the average PSNR is reported to measure the playback video quality.

Figure 3 shows the performance of both ERDBS and ERDBS-H schemes under different values of RTT, where the start-up delay is 200ms and the mean packet-loss rate $\epsilon$ is 20%. As can be seen, when RTT is less than 90ms, the ERDBS-H is inferior to ERDBS and the discrepancy is around $0.8dB$. This is mainly due to the redundancy introduced by FEC in ERDBS-H. When RTT increases, ERDBS-H outperforms ERDBS.

Figure 4 illustrates the performance of the two scheduling systems under different start-up delays, where the RTT is 100ms and the mean packet-loss rate is 20%. As shown in the figure, ERDBS-H outperforms ERDBS when the start-up delay is less than 600ms. This is because with a smaller start-up delay, the packet loss recovery capability of ARQ is rather limited. In contrast, ERDBS-H is able to take advantage of FEC and reacts much faster to recover packet loss.

Finally, the performance of the schemes under different mean

$$P\left(l_{u,v}\right)=\begin{cases}1, & \text{if } l_{u,v} \text{ is ACKed;}\\ \left(1-\epsilon^{n}\right)+\epsilon^{n}\cdot p\left(k-n^{ACK},n^{OUT}-1\right), & \text{if } l_{u,v} \text{ has been sent out } n \text{ times without ACK;}\end{cases} \tag{6}$$

$$p\left(k-n^{ACK},n^{OUT}-1\right)=\begin{cases}0, & \text{if } k-n^{ACK}\le n^{OUT}-1;\\ \sum\limits_{i=k-n^{ACK}}^{n^{OUT}-1}\begin{pmatrix} n^{OUT}-1\\ i\end{pmatrix}\cdot(1-\epsilon)^{i}\cdot\epsilon^{n^{OUT}-i}, & \text{else.}\end{cases} \tag{7}$$

$$\hat{d}_{FEC}=\begin{cases}\begin{pmatrix} n^{OUT}\\ k-1-n^{ACK}\end{pmatrix}\cdot(1-\epsilon)^{k-1-n^{ACK}}\cdot\epsilon^{n^{OUT}-\left(k-1-n^{ACK}\right)}\cdot\sum\limits_{l_{u,v}\in G_{NACK}}\hat{d}_{u,v}, & \text{if } k-1-n^{ACK}\le n^{OUT};\\ 0, & \text{else;}\end{cases} \tag{9}$$

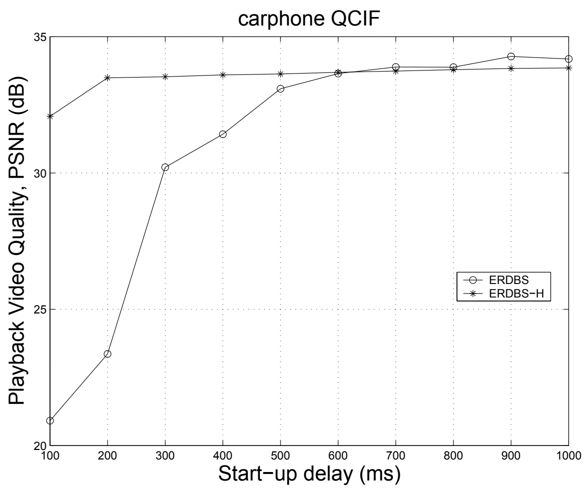


**Fig. 4**. Performance of the scheduling schemes under different start-up delays. RTT is 100ms and the mean packet loss rate is 20%.

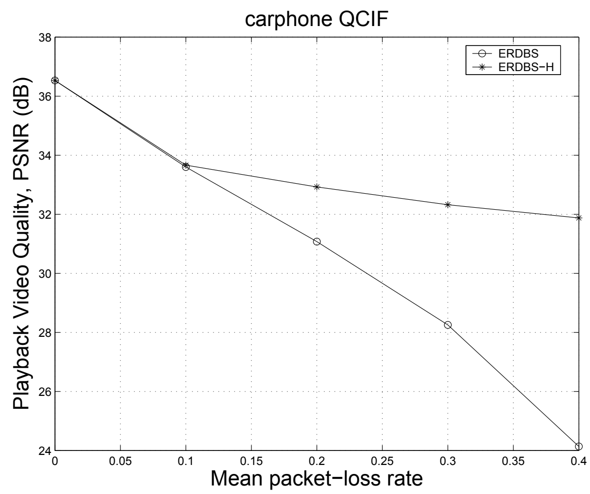


**Fig. 5**. Performance of the scheduling schemes under different mean packet-loss rates. RTT is 100ms and the start-up delay is 200ms.

packet-loss rates is depicted in Figure 5, which clearly shows the error robustness of our proposed scheme.

## 5. CONCLUSION

In this paper, we propose a transmission scheme with hybrid error control for streaming scalable video across the packet erasure networks. In contrast to the conventional ERDBS scheme [4] which only adopts ARQ for packet loss recovery, our proposed ERDBS-H scheme utilizes both FEC and ARQ to improve the robustness. Experimental results show that our proposed ERDBS-H scheme is able to outperform the original ERDBS scheme under larger round-trip-time and/or smaller start-up delays. Also, the hybrid error control scheme is more robust to higher packet-loss rates.

## 6. REFERENCES


[1] H. Radha, M. Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Trans. Multimedia*, vol. 3, pp. 53-68, Mar. 2001.

[2] T. Stockhammer, H. Jenkac, and C. Weiss, "Feedback and error protection strategies for wireless progressive video transmission," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, pp. 465-482, Jun. 2002.

[3] J. Chakareski, P. Chou, and B. Aazhang, "Computing rate-distortion optimized policies for streaming media to wireless clients," in *Proc. IEEE DCC'02*, Snowbird, UT, Apr. 2002, pp. 53-62..

[4] Z. Miao and A. Ortega, "Expected run-time distortion based scheduling for delivery of scalable media," *Proc. Packet Video*, Pittsburg, PA, Apr. 2002.

[5] P. A. Chou and Z. Miao, "Rate-distortion optimized sender-driven streaming over best-effort networks," *Proc. IEEE Workshop on Multimedia Signal Processing*, Cannes, France, pp. 587-592, Oct. 2001.