Real-Time Video Streaming Using Randomized Expanding Reed-Solomon Code

Jimin Xiao, Tammam Tillo, *Senior Member, IEEE*, Yao Zhao, *Senior Member, IEEE*

***Abstract*—Forward error correction (FEC) codes are widely studied to protect streamed video over unreliable networks. Typically, enlarging the FEC coding block size can improve the error correction performance. For video streaming applications, this could be implemented by grouping more than one video frame into one FEC coding block. However, in this case, it leads to decoding delay, which is not tolerable for real-time video streaming applications. In this paper, to solve this dilemma, a real-time video streaming scheme using randomized expanding Reed-Solomon code is proposed. In this scheme, the Reed-Solomon coding block includes not only the video packets of the current frame, but could also include all the video packets of previous frames in the current group of pictures. At the decoding side, the parity-check equations of the current frame are jointly solved with all the parity-check equations of the previous frames. Since video packets of the following frames are not encompassed in the RS coding block, no delay will be caused for waiting for the video or parity packets of the following frames both at encoding and decoding sides. Experimental results show that the proposed scheme outperforms other real-time error resilient video streaming approaches significantly, specifically, for the Foreman sequence, the proposed scheme could provide 1.5 dB average gain over the state-of-the-art approach for 10% i.i.d packet loss rate, whereas for the burst loss case, the average gain is more than 3 dB [1].**

***Index Terms*—video streaming, randomized expanding Reed-Solomon, forward error correction, real-time, error resilient**

I. Introduction

REAL-TIME video steaming in lossy network environments, whether it is wireless or wired, is a challenging task. High compression ratio video coding, which gains from the prediction based motion estimation and entropy coding, makes the coded streams sensitive to transmission losses. In fact, distortions caused in one frame or even a portion of a frame will propagate to the following frames, which will result in serious degradation of the reconstructed video quality [1], [2]. Therefore, the study of error resilient techniques for video coding and video streaming is an important task.

There have been many error resilient techniques proposed to meet the requirements of video streaming over unreliable networks, and an overview of error resilient techniques is presented in [3]. One category of error resilient techniques uses intra macroblock (MB) refreshment [4], [5] to stop error propagations; and later in [6], [7], not only the intra coding modes but also the motion prediction paths are optimally chosen to minimize the error propagations by taking the lossy network conditions into consideration. One advantage of these approaches is that no delay will be caused [2], making them suitable for real-time video streaming applications. However, the coding efficiency of intra mode is much lower than inter mode, therefore the overall coding efficiency is compromised, nevertheless in [8], it is reported that if intra coding is twisted with redundant coding in a good manner, the error resilient performance could be improved. Another category of error resilient video streaming techniques are based on feedback information, including Automatic Repeat reQuest (ARQ) [9]–[11] and feedback-based Reference Picture Selection (RPS) [12], [13]. These techniques usually cause long delay because of the network round-trip time, and consequently they cannot be employed for real-time applications. The third category of error resilient techniques use the concept of redundant picture/slice coding with equal or lower quality [8], [14], [15], which was used in multiple description coding (MDC) [16], [17]. For the redundant picture/slice coding and MDC schemes, usually no delay is caused, but when the redundant version is used to replace the primary one or some of the descriptions are lost during transmission, there will be mismatch error and it will propagate all over the group of pictures (GOP). Whereas for the approaches that use the forward error correction (FEC) coding with unequal loss protection [18], [20], [21], the delay depends on the FEC coding block size. In fact, the need to wait for the whole FEC coding block in order to recover a lost packet means that an extra delay will be introduced. In [18], the Reed-Solomon (RS) coding block includes the whole GOP, and one GOP of delay is caused. Another work that use unequal error protection based on frame dependency is [19], where the encoding unit includes all the data of one GOP, so one GOP of delay is caused. In [20], the RS coding block contains one block of packets (BOP) generated from different frames, and unequal loss protections are allocated for different packets based on both the frame position in the GOP and the data partition it belongs to. For this approach, one

This work was supported by National Natural Science Foundation of China (No.60972085, No.61210006), the National Science Foundation of China for Distinguished Young Scholars (No.61025013), and National Basic Research Program of China (Grant No. 2012CB316400).

Jimin Xiao is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3GJ, UK and the Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, 111 Ren Ai Road, Suzhou, P.R. China.

Tammam Tillo is with the Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, 111 Ren Ai Road, Suzhou, P.R. China, E-mail: (tammam.tillo@xjtlu.edu.cn)

Yao Zhao is with Institute of Information Science, Beijing Jiaotong University, Beijing Key Laboratory of Advanced Information Science and Network Technology, Beijing, China.

[1]Matlab code of this work is available for download at http://www.mmtlab.com

[2]With the term *delay*, we refer to the elapsed time between receiving packets of a frame (in the first attempt of sending) and the time the video decoder starts decoding them at the receiver side.

BOP of delay is caused, i.e., the delay depends on the length of the BOP. In [21], the RS code is implemented at frame level using adaptive slice grouping and unequal error protection (UEP), and no FEC coding delay is created. However, for the frame level FEC approach, usually the quantity of the source packets is not large enough for the FEC code to be efficient. Whereas in [22] a cross-layer adaptive error protection scheme is proposed, where MAC-layer transmission of real-time video over wireless local area networks (WLANs) is optimized using cross-layer optimization, and in [23], the authors used a slice loss visibility (SLV) model to optimize video transmission over an orthogonal frequency division multiplexing (OFDM) system.

For the forward error correction coding schemes, the error correction performance and the FEC decoding delay are two contradicting requirements. On one hand, enlarging the FEC coding block size, i.e., grouping video packets from more than one video frame, can improve the error correction performance, however, this will cause some delay equivalent to the length of the FEC coding block. On the other hand, small FEC coding block size, i.e., FEC implemented at frame level, will cause no delay for waiting for the video or parity packets of the following frames, however, the error correction performance is compromised. To solve this problem, in our previous work [24], [25], we proposed the Dynamic Sub-GOP FEC Coding (DSGF) approach, where the Sub-GOP, which contains video packets of more than one video frame, is used as the FEC coding block. It is found that in the DSGF approach, FEC codes can stop error propagations efficiently, thereby it could provide better overall video quality than frame level FEC, yet no delay will be caused for waiting for the video or parity packets of the following frames. However, in [24], [25], the FEC error correction capability can only be used by the last frame of each Sub-GOP, and consequently the video quality could fluctuate.

To overcome the previous challenges, in this paper, a Randomized Expanding Reed-Solomon (RE-RS) scheme is proposed for real-time video streaming applications. In the proposed RE-RS scheme, RS parity packets are allocated for each frame of the GOP. They are generated using the video packets of the current frame and all the previous frames of the current GOP. At the decoder side, the parity-check equations of the current frame will be combined with those of all the previous frames. The lost packets will be recovered if the combination of the parity-check equations can be jointly solved. Thereby, these RS parity packets will not only help to recover the lost packets of the current frame, but also the lost packets of the previous frames. It is worth noticing that recovering the lost packets of the previous frames will not affect their timely decoding and visualization. In fact, during their decoding time they will be concealed and displayed, so their later recovering will help reducing the propagated errors. In this scheme, no video packets of the following frames will be used in the current RS coding block, thereby each frame could be decoded and displayed at its display time, and no extra delay will be needed to wait for the source and parity packets of the following frames. Moreover, for the RE-RS scheme, there will be no frame-by-frame video quality fluctuation problem that affects the DSGF [25] approach, and more in general Sub-GOP based approaches. It is worth mentioning that sliding window [26] and expanding window [27], [28] fountain code protection for SVC [29] have already been proposed, where they target layered hierarchical data, i.e., SVC. To the best of our knowledge, twisting the expanding window RS code with the reference updating technique for real-time steaming of non-layered data, i.e., H.264/AVC date, is novel. In [26]–[28], the LT or Raptor code was used. However, with the introduction of the RaptorQ code [30] the performance of these methods would be improved, given that RaptorQ code requires less overhead.

The contribution of this paper is many fold: firstly, the expanding window RS code is introduced in combination with the reference buffer updating technique for real-time video streaming applications. Secondly, in order to ensure that the equations of different windows could be jointly RS decoded, so as to increase the probability of recovering current and previous losses, we proposed a randomized RS code for the expanding window approach. Thirdly, we investigated the parity allocation problem in the new paradigm of expanding window RS code for video data, and it has been found that evenly allocating the parity packets among frames is a simple yet efficient method. Fourthly, a simplified sliding window scheme is proposed to lower the computational complexity and the memory requirement without compromising its error resilient performance too much.

The rest of the paper is organized as follows. A brief review of systematic RS code is provided in Section II. In Section III the proposed RE-RS scheme is presented in detail. In Section IV some experimental results validating the proposed approach are given. Finally, some conclusions are drawn in Section V.

II. Systematic Reed-Solomon Erasure Code

In this section, we will recall some concepts and theory about systematic Reed-Solomon (RS) erasure code, which will be used throughout this paper. The systematic RS erasure code has been widely studied as application layer FEC code to protect data packets against losses in packet erasure networks. In RS (N, K) code, for every K source packets, $N-K$ parity packets are generated to make up a codeword of packets, with a total length of N packets. As long as a client receives at least K out of the N packets, it can recover all the source packets. If the received packet number is less than K, the received source packets can still be used, because they have been kept intact by the systematic RS encoding process. For the RS (N, K) code, the N and K could be any positive integer under the following constraint:

$$\begin{cases} N \leq 2^m - 1 \\ K < N \end{cases} \tag{1}$$

where m is the number of bits in a symbol. When $N < 2^m - 1$, it is referred to as the short form of the code. In this case, $2^m - 1 - N$ zero padding packets are added to the K source packets, which makes the total number of packets $2^m - 1 + K - N$ before RS coding, we will call this the full length source code. The RS code will add $N - K$ parity packets, which makes the total packet number $2^m - 1$. After encoding, the padded

zero packets are removed to form a so called shortened Reed-Solomon code, whereas at the decoder side these zeros are re-inserted. Let us assume the systematic RS code is $C = (c_1, c_2, \ldots, c_n)$ where $n = 2^m - 1$, and among them t symbols are lost at the decoder side with their indexes being i_1, i_2, …, i_t. Thus, $c_{i_1} = X_1$, $c_{i_2} = X_2$, …, $c_{i_t} = X_t$ are the t lost variables. The decoder will try to recover the lost packets by solving the parity-check equations:

$$CH^T = 0 \tag{2}$$

where H is the parity-check matrix, and it could be denoted as follows:

$$H = \begin{bmatrix} 1 & \alpha & \ldots & \alpha^{2^m-3} & \alpha^{2^m-2} \\ 1 & \alpha^2 & \ldots & (\alpha^2)^{2^m-3} & (\alpha^2)^{2^m-2} \\ \vdots & \vdots & & \vdots & \vdots \\ 1 & \alpha^{N-K} & \ldots & (\alpha^{N-K})^{2^m-3} & (\alpha^{N-K})^{2^m-2} \end{bmatrix} \tag{3}$$

with α being the primitive element of Galois Field $GF(2^m)$. Since H is a full rank matrix and its rank is $N - K$, so (2) could be solved when the variable number is not more than $N - K$, which also means that it can recover up to $N - K$ erased symbols.

III. Randomized Expanding Reed-Solomon Scheme

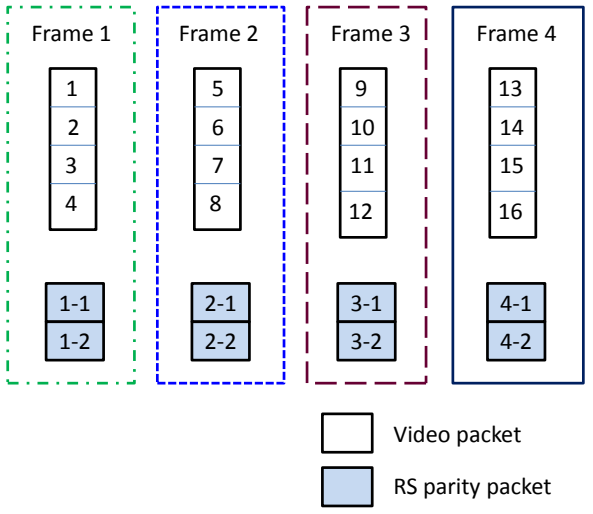
For real-time FEC video streaming applications, one common approach is to perform RS coding in frame level, which means that the RS coding block contains video packets from the same video frame. Under this constraint, to recover the lost packets, the RS decoder does not need to collect source packets of many frames, therefore there will be no decoding delay. To analyze this approach, that will be used for comparison, let us assume the GOP length is L frames, and the i-th frame has $S(i)$ source packets and $R(i)$ RS parity packets. If we want to have even distribution of the parity packets among all the GOP frames, then $R(i)/S(i)$ needs to be almost constant over all the GOP's frames. In general, the number of generated slices per frame, $S(i)$, varies from frame to frame due to different level of motion level and texture complexity, and because the number of parity packets to be inserted, $R(i)$, should be integer, so we can write $R(i)$ as following:

$$R(i) = \begin{cases} \lceil \mu S(1) \rceil & \text{if } i == 1 \\ \left\lceil \mu \sum_{k=1}^{i} S(k) \right\rceil - \sum_{k=1}^{i-1} R(k) & \text{if } i > 1 \end{cases} \tag{4}$$

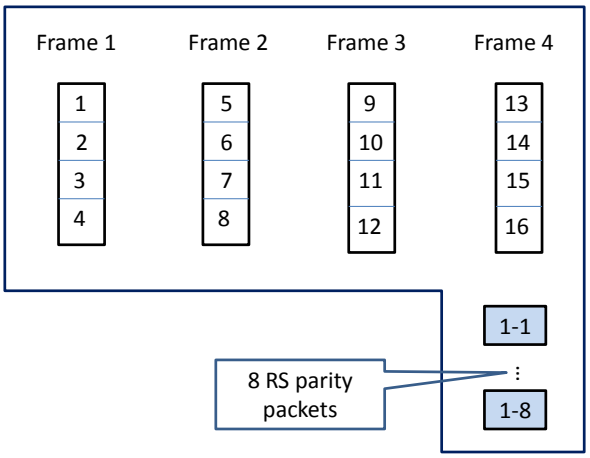
where $\mu = (N - K)/K$ is the redundant packet rate of RS code, and operation $\lceil X \rceil$ is used to get the minimum integer number greater than or equal to X. In this case, using formula (4) makes the average inserted redundant packet rate among several frames approach μ. Since in this approach, the RS parity packets are almost evenly allocated among all the video frames, so this will be called Evenly FEC in the following. An simplified example of the Evenly FEC is shown in Figure.1-(a), where for $\forall i$ $S(i) = 4$, and $\mu = 0.5$.

For the Evenly FEC, there are two fundamental problems that make its error correction performance low:

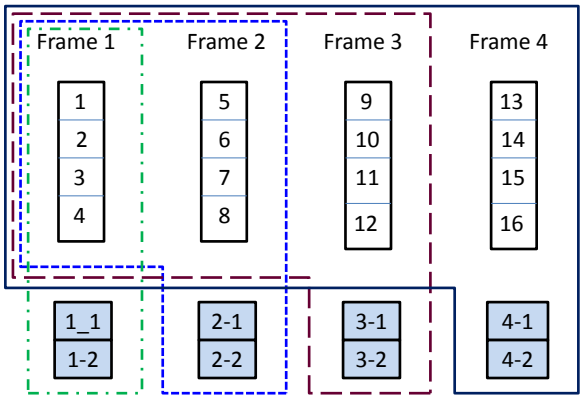
1) The number of video source packets generated in each frame is small, which makes the RS code inefficient. To



(a) Evenly FEC



(b) Sub-GOP based FEC



(c) proposed RE-RS scheme

Figure 1. Examples of different FEC schemes, where each frame has 4 video packets and redundant packet rate is $\mu = 0.5$. (a) Evenly FEC; (b) Sub-GOP based FEC; (c) proposed RE-RS scheme.

Table I
THE REMAINING PACKET LOSS RATE AFTER RS CODE CORRECTION WITH $\mu = 0.2$, WHERE RS CODING BLOCK SIZE IS $K = 5, 10, 15, 20, 30$, NETWORK PACKET LOSS RATE IS $p = 5\%, 10\%, 15\%$

p	$K=5$	$K=10$	$K=15$	$K=20$	$K=30$
5%	1.13%	0.51%	0.25%	0.13%	0.04%
10%	4.10%	3.03%	2.38%	1.93%	1.32%
15%	8.34%	7.62%	7.20%	6.91%	6.47%

better visualize the effects of K, the number of source packets, on the error correction capability of the RS code, Table I lists the remaining packet loss rate, p', after the RS code correction, for different values of K. This table demonstrates that for the same packet loss rate and redundancy, the smaller the RS coding block the lower the performance of the RS codes.

2) The error correction capability of the current frame cannot help to recover the lost packets of the previous frames, which may cause the distortion of the unrecovered packets in the previous frames propagate to the current and following frames. Let us take the first two frames in Figure.1-(a) as an example. Let us assume three video packets in the first P-frame are lost, while other packets are received. In this case, the RS code of the first P-frame will fail to recover the three lost packets. Meanwhile, as there is no packet loss in the second frame, its error correction capability will not be exploited, and it cannot be used to help recovering the previous losses in the first P-frame. Whereas if this error correction capability could be exploited to help to recover the lost packets in the previous frames, then it might be possible to recover the three lost packets, and thereby the propagation distortion from the previous frames will be reduced.

In order to overcome the above problems of the Evenly FEC scheme, we could come out with two possible solutions that are described as follows.

A. Sub-GOP Based FEC Scheme

One solution is grouping video packets of one Sub-GOP, which contains more than one video frame, into one RS coding block. Figure.1-(b) shows one example for this case. By doing this, the K value of the RS code will be increased, and consequently this will improve the performance of the RS code. However, for this solution, one Sub-GOP of delay will be caused if the video frames will be decoded and displayed at the end of its Sub-GOP. On the other hand, if the Sub-GOP based approach is to be used for the real-time applications where it is not possible to wait for the video or parity packets of the following frames, the average video quality might deteriorate with respect to the approach that tolerates delay. In [25], we proposed to decode and display the frames in real-time fashion, and at the end of each Sub-GOP, if the lost packets could be recovered by the RS code, the reference buffer will be updated using the recovered information to stop the error propagations. In this scheme, although the overall performance could be higher than that of the Evenly FEC, the video quality fluctuates frame by frame.

B. Proposed RE-RS scheme

Another solution which we propose in this paper is to use expanding RS code. This scheme is described in Figure.1-(c), the RS parity packets are generated using the video packets of the current frame and all the previous frames of the current GOP. At the decoder side, the parity-check equations of the current frame will be combined with those of all the previous frames. The lost packets will be recovered if the combination of the parity-check equations can be jointly solved. Thereby, these RS parity packets can not only help to recover the lost packets of the current frame, but also the lost packets of the previous frames. In this scheme, no video packets of the following frames will be used in the current FEC coding block, thereby each frame could be decoded and displayed at its proper time, and no extra delay will be caused.

To better illustrate the expanding Reed-Solomon scheme, one simplified example is drawn. Let us use the first two frames in Figure 1-(c), where each frame has 4 video packets and 2 RS parity packets. We assume packets 1, 2 and 3 in the first frame and packet 5 in the second frame are lost, whereas other video packets and parity packets of the two frames are received. In this case, as described in (2), the parity-check equations for the first frame could be simplified as following:

$$\begin{cases} X_1 + \alpha X_2 + \alpha^2 X_3 + C_1 = 0 \\ X_1 + (\alpha^2) X_2 + (\alpha^2)^2 X_3 + C_2 = 0 \end{cases} \tag{5}$$

and the parity-check equations for the second frame combined with that of the first frame are as following:

$$\begin{cases} X_1 + \alpha X_2 + \alpha^2 X_3 + C_1 = 0 \\ X_1 + (\alpha^2) X_2 + (\alpha^2)^2 X_3 + C_2 = 0 \\ X_1 + \alpha X_2 + \alpha^2 X_3 + \alpha^4 X_5 + C_3 = 0 \\ X_1 + (\alpha^2) X_2 + (\alpha^2)^2 X_3 + (\alpha^2)^4 X_5 + C_4 = 0 \end{cases} \tag{6}$$

where X_1, X_2, X_3 and X_5 denote the four lost packets; C_1, C_2, C_3 and C_4 are constant values, which are determined by the received packets. It should be mentioned that, in order to recover the lost packets, the above equations should be solved in Galois Field of $GF(2^m)$. Here we should note that in the first and third equations in (6), the coefficients for X_1, X_2, X_3 are the same, and this also happens for the second and fourth equations. Therefore, for (5) and (6), the rank of coefficient matrix is 2 and 3, respectively. Hence, (5) and (6) cannot be solved with three and four variables, respectively. In other words, all the four lost packets, in this example, cannot be recovered.

In order to tackle this problem, we need to increase the rank of (6) to 4, for this reason we propose to randomly reorder the video packets and the zero padded packets (if any) before the RS encoding stage. This is a key step to ensure that, with high probability the coefficients of the parity-check equations for different frames are independent, thereby ensure the high error correction performance of the proposed RE-RS scheme. Let us see an example where we assume that each symbol has 4 bits, or in other words, the Galois Field is $GF(2^4)$, and RS code $(16, 14)$ is used [3], which means that if we have only 4 source

[3]This means that at maximal three frames could be protected in this case study.

packets, 10 zero packets will be added before generating the 2 parity packets to make the full length RS code. Let us assume for the first RS window, the randomly reordering positions for the lost packets $\{1, 2, 3\}$ become $\{6, 3, 11\}$; whereas for the second RS window the randomly reordering positions for the lost packets $\{1, 2, 3, 5\}$ become $\{7, 1, 4, 12\}$. In this case, the combined RS parity-check equations for the second frame becomes:

$$\begin{cases} \alpha^5 X_1 + \alpha^2 X_2 + \alpha^{10} X_3 + {C_1}' = 0 \\ (\alpha^2)^5 X_1 + (\alpha^2)^2 X_2 + (\alpha^2)^{10} X_3 + {C_2}' = 0 \\ \alpha^6 X_1 + X_2 + \alpha^3 X_3 + \alpha^{11} X_5 + {C_3}' = 0 \\ (\alpha^2)^6 X_1 + X_2 + (\alpha^2)^3 X_3 + (\alpha^2)^{11} X_5 + {C_4}' = 0 \end{cases} \quad (7)$$

In the Galois Field $GF(2^4)$, the coefficients matrix for (7) is

$$\begin{bmatrix} \alpha^5 & \alpha^2 & \alpha^{10} & 0 \\ \alpha^{10} & \alpha^4 & \alpha^{20} & 0 \\ \alpha^6 & 1 & \alpha^3 & \alpha^{11} \\ \alpha^{12} & 1 & \alpha^6 & \alpha^{22} \end{bmatrix} = \begin{bmatrix} \alpha^5 & \alpha^2 & \alpha^{10} & 0 \\ \alpha^{10} & \alpha^4 & \alpha^5 & 0 \\ \alpha^6 & 1 & \alpha^3 & \alpha^{11} \\ \alpha^{12} & 1 & \alpha^6 & \alpha^7 \end{bmatrix}.$$

It is worth noticing that the rank of this matrix is 4, so it is full rank matrix, and the equations can be solved. Therefore, by the second frame all the four lost packets can be recovered.

C. Detailed Procedure of RE-RS Scheme

Since our objective is to design FEC video transmission system for real-time applications while minimizing the delay caused by the encoding stage, therefore B-frame will not be used, so the IPPP GOP structure will be used. This choice is also justified by the fact that video telephony, the most commonly used applications for real-time system, typically uses the baseline profile of H.264/AVC, where only I-frames and P-frames are used [31]. To make the RS code efficient, fixed length slice scheme in term of byte, will be used to create slices. In this method, the macroblocks in each frame will be scanned in raster-scan order and encapsulated into slices with the constraint that the size of each slice should not be more than the target length of the slices, therefore, the length of all the slices except the last ones in each frame will be very close to the target length. Whereas the last slice in each frame will be in general smaller than the target length. Thus, for slices other than the last one, only very few zero bytes are padded to reach the target packet length. Whereas for the last slice in each frame, usually more dummy zero bytes are used for padding. It is important to note that, in the proposed scheme, the length of the packets used to encapsulate the RS parity symbols is similar to the length of the packets used to encapsulate video slice data, and this latter is dictated by the Maximum Transmission Unit (MTU) of the underlying networks. So consequently, throughout this paper, the term packet and slice are used interchangeably, as one packet per slice packetization method is adopted.

The RE-RS procedure at the video sender side works as following:

1) RS parity packets are allocated for each frame in the GOP using (4); that means the redundancy is evenly distributed among the GOP frames, and $R(i)$ is the amount of parity packets inserted for the i-th frame.
2) Video packets of the current frame and all the previous frames of the current GOP are collected. If the total video packets number is less than $2^m - 1 - R(i)$, zero padding is used. All video packets are ordered as they are generated by the H.264/AVC video encoder, where the zero padding packets are appended after the video packets. Let us take the second RS window in Figure 1-(c) for example, the order of the video packets is: 1, 2, 3, 4, 5, 6, 7, 8.
3) The $2^m - 1 - R(i)$ packets are randomly reordered. Let us assume that for the i-th frame, the new position for the k-th packet is $O_i(k)$, and $O_i(k)$ should meet the following requirements:
$$\begin{cases} O_i(k) \in [1, 2^m - 1 - R(i)] \\ O_i(k_1) \neq O_i(k_2), \forall\ k_1 \neq k_2 \end{cases} \quad (8)$$
Here it is important to note that for different frames, different reordering maps should be used. Moreover, in order to make the decoder work properly, the sender and receiver should have the same maps.
4) $R(i)$ RS parity packets are generated using the reordered $2^m - 1 - R(i)$ video packets and zero padded packets (if any). Taking the second RS window in Figure 1-(c) as example, the generated parity packets are 2-1 and 2-2.
5) Together with the video packets of the current frame, $R(i)$ RS parity packets are transmitted to the receiver side.
6) Repeat steps 2-5 for all the video frames in the current GOP.

At the video receiver side, all the received packets of the previous frames in the current GOP will be kept in the buffer, and the decoding procedure of the RE-RS scheme will work as following:

1) The receiver will collect the video packets of the current frame, and together with the previously received and buffered packets of the current GOP, they will be reordered using the same reordering map used at the video sender side.
2) By multiplying the reordered video packets with the parity-check matrix, the parity-check equations are generated, as in (2), the parity-check equations for the current frame include $R(i)$ equations, and they will be kept for the RS decoding of the following frames in this GOP.
3) The parity-check equations of the current frame are combined with all the parity-check equations of the previous frames of the current GOP. The combined equations for the i-th frame will include $\sum_{k=1}^{i} R(k)$ equations.
4) If the combined equations could be solved, and consequently if it allows to recover some of the non-recovered packets in the previous frames, then the frames that these packets belong to and the following frames will be video re-decoded with all the recovered packets, and the reference buffer will be updated with the newly decoded information. It is worth mentioning that the reference buffer updating technique was also used to exploit the

late arrival packets to stop error propagations in [32], [33]. If the combined equations cannot be solved, then the current video frame will be video decoded with all the received video packets and the non-arrived slices will be concealed, then the reference buffer will be updated.

5) Repeat all the above process for all the video frames in one GOP.

D. Performance of Randomly Reordering

The performance of the proposed RE-RS scheme critically depends on removing the linear dependency between the parity-check equations by randomly reordering the source packets. In order to show that the randomly reordering is good for this task, let us take a case study, where the parity packet number for each frame is 1, the lost packet number in the first frame is n, and in the following $n-1$ frames, there is no packet loss, which means that their parity packets will be used to recover the lost packets in the first frame. We assume that for the expanding window of the u-th frame, the position of the v-th lost packet after randomly reordering becomes $i_{u,v}$ ($1 \leq u \leq n$, $1 \leq v \leq n$) with $1 \leq i_{u,v} \leq 2^m - 1$, then the coefficient matrix of the combined parity-check equations of the first n frames is:

$$\begin{bmatrix} \alpha^{i_{1,1}-1} & \alpha^{i_{1,2}-1} & \dots & \alpha^{i_{1,n}-1} \\ \alpha^{i_{2,1}-1} & \alpha^{i_{2,2}-1} & \dots & \alpha^{i_{2,n}-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha^{i_{n,1}-1} & \alpha^{i_{n,2}-1} & \dots & \alpha^{i_{n,n}-1} \end{bmatrix}. \quad (9)$$

Typically $n \ll 2^m - 1$, which means that the elements of the generated matrix by the randomly reordering process could be regarded as being i.i.d selected from the Galois Field $GF(2^m)$. According to [34], the probability that this matrix is full rank is $\prod_{i=1}^{n} \left(1 - (2^m - 1)^{-i}\right)$. So for example with $m = 8$ and $n \in [1, 10]$, this probability is almost 0.9961. From this we could conclude that using the randomly reordering process can remove the linear dependency between the parity-check equations with high probability.

Based on the above finding, we could conclude that it is reasonable to assume that the randomization process achieves its objective, and this will also be demonstrated by the results obtained with the video sequences in the experimental section. From now on, we will assume that the combined parity-check equations are linearly independent, this means that if we have "proper" redundant packet rate, all the lost packets could be recovered by waiting for several frames so as to accumulate enough parity packets to solve the equations. In the following, we will study the time interval needed to wait so as to recover all the lost packets. To do this, let i represent the index of the current frame and $d(k)$ to denote the lost parity and video packet number for frame k, with $1 \leq k \leq i$. To recover all the lost video packets by the decoding time of the i-th frame, the set $\Phi = \{d(k), k \in [1, i]\}$ should satisfy the constraint:

$$C(i) = \{\Phi | \sum_{j=0}^{t} d(i-j) \leq \sum_{j=0}^{t} R(i-j), \forall t \in [0, i-1]\} \quad (10)$$

where $R(k)$ is the number of RS parity packets for the k-th frame as previously defined. The reason behind this equation is two fold, the first is that for $\forall t \in [0, i-1]$ the lost packets among frames $[i-t, i]$ could only be recovered by using the RS parity packets allocated for these frames, but not the previous parity packets, i.e., those frames before the $(i-t)$-th frame; second, the number of the lost packets should be less than the allocated RS parity packets among frames $[i-t, i]$. Later on, the error correction capability of the following frames could also be used to recover the lost packets in frames $[1, i]$, thereby, all the lost packets in frames $[1, i]$ could be recovered by the time of decoding the j-th frame with $j > i$, if the following condition is satisfied:

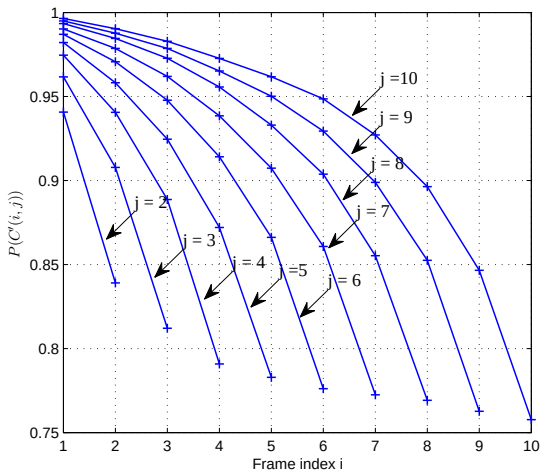
$$C'(i, j) = C(i) \bigcup C(i+1) \dots \bigcup C(j). \quad (11)$$

This is because for $\forall k \in [i, j]$, $C(k)$ insures that all the lost packets among frames $[1, k]$ could be recovered. To numerically evaluate the upper bound performance of the randomization process for parameter setting $\{S, \mu, p\}$, let us define the probability that the set $\{d(k), k \in [1, j]\}$ meets the constraint $C'(i, j)$ by $P(C'(i, j))$. At this point now let us evaluate the probability $P(C'(i, j))$ for a few cases, where the slice number per frame, redundant packet rate and the i.i.d average packet loss rate $\{S, \mu, p\}$ are $\{5, 0.2, 0.1\}$ and $\{10, 0.2, 0.05\}$, and 10000 trials have been carried out for 10 frames. Figure 2 shows the value of $P(C'(i, j))$ for this simulation. It is observed that for the same value of i, the larger the value of j is, the higher probability $P(C'(i, j))$ could be. This is because for larger j, there are more RS parity packets in the following frames that could help to recover the lost packets among frames $[1, i]$. Moreover, we could notice in Figure.2-(b), where the average packet loss rate is relatively small in comparison with the redundant packet rate, and a large number of packets per frame are generated, it is almost certain that all the lost packets in previous frames $[1, j-3]$ can be recovered by frame j. Whereas, in Figure.2-(a) the average packet loss rate is relatively high and the number of packet per frame is small, then more time is needed to fully recover the lost packets.

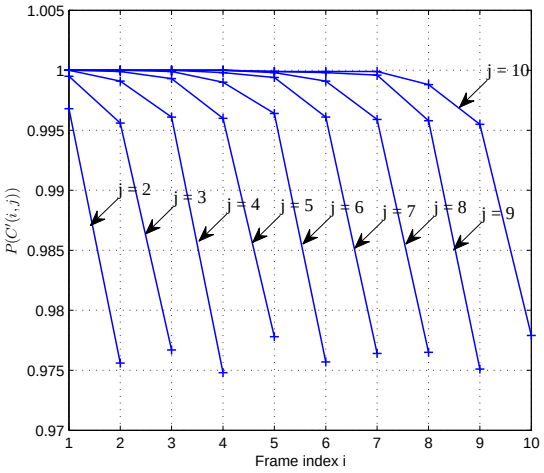
From practical point of view, it should be mentioned that, sending the reordering maps to the receiver side costs some bitrate. Thus, the same reordering maps could be used for different GOPs. In this case, the overall bitrate cost of sending the reordering maps could be neglected.

E. Why Evenly Allocating Parity Packets

As described in Section.III, in the proposed RE-RS scheme, the allocated parity packets are evenly distributed among all the frames using (4). In this section, the reason for allocating the parity packets in this fashion will be explained. To simplify the problem, some assumptions will be used: each P-frame has the same number of slices; the mismatch distortion caused by losing each slice is the same; and proper redundant packet rate is used, which means that after certain number of frames, all the lost packets could be recovered. At this point, let us assume a hypothetical scenario in which some packets are lost among frames $[i_1, i_1+t]$ whereas other packets of the GOP are received intact. Given the assumption that a proper amount of redundancy is inserted, these lost packets could be recovered



(a)



(b)

Figure 2. The value of $P(C'(i,j))$ for different i, j; (a) each frame has 5 slices and 1 parity packet, packet loss model is i.i.d, with packet loss rate 10%; (b) each frame has 10 slices and 2 parity packet, packet loss model is i.i.d, with packet loss rate 5%.

after w frames. This means that the concealment distortion and propagated distortion will affect the frames $[i_1, i_1+t+w-1]$. If now hypothetically we assume that the same pattern of errors affects frames $[i_2, i_2+t]$, $i_2 \neq i_1$, and we want to insert a certain amount of redundancy to recover these losses, then a question will rise: whether $[i_1, i_1+t]$ or $[i_2, i_2+t]$ should be protected more? To answer this, if we take the previous assumptions into consideration, i.e., the frames $[i_1, i_1+t]$ and $[i_2, i_2+t]$ have the same number of slices, and each slice lose leads to the same amount of mismatch distortion, and the two groups of frames are randomly chosen, then we could conclude that neither $[i_1, i_1+t]$ nor $[i_2, i_2+t]$ should be favored in terms of redundancy, therefore the two groups of frames should be treated equally. In other words, the two groups of frames should have the same pattern of redundancy. So now if we generalize this for different t, then we reach a further conclusion that the pattern of redundancy should be uniform. In Figure 2.(b) we could see that having uniform redundancy will lead to constant error propagation window, in other words, if $i \leq j-3$, then $P(C'(i,j)) \approx 1$. This means that the distortion will only propagate for no more than 3 frames no matter the frame position within one GOP is.

F. Sliding Window RS code: A Simplified Solution

The full version of the proposed scheme requires storing all the video slices and parity packets, so in order to lower the computational complexity and the memory requirement for both the RS encoding/decoding and the reference updating process, one simplified scheme is proposed, where instead of using the expanding window RS code, sliding window RS code is adopted. In other words, the RS parity packets will be generated using the video packets of the current frame and several frames before the current frame, for example, W frames, where W refers to the sliding window size. Accordingly, at the decoder side, the parity packets in the current frame can help to recover the lost packets within its window. The sliding window scheme is based on the assumption that when proper among of parity packets are inserted, for the current frame i, with high probability all the lost packets before this window, i.e., before frame $i-W+1$, are already recovered, so whether or not the RS coding block includes packets before frame $i-W+1$ will make no difference. Otherwise, it will fail to recover the lost packets within this window. So this scheme will sacrifice the error resilient performance slightly. For example, for the reported case in Figure 2.(b), if a sliding window approach is used, with $W \geq 3$, then the performance will not be sacrificed too much. Moreover, the experimental results reported in Section IV also show that if proper sliding window size is used, its performance gap in comparison with full expanding window approach is not big.

In the proposed system, the decoding process differs from conventional systematic RS decoding, due to the use of re-ordering of source packets. It also involves joint decoding/error correction of codeword packets of both the current and prior frames. So the fast algorithm of decoder implementation will be essential for power-constrained devices, and this work is left for future research.

IV. Experimental Results

Our experimental setting is built on the JM14.0 [35] H.264/AVC codec [4]. CIF video sequence Paris, Foreman, Bus, Stefan and Mobile are used for the simulations. We selected these sequences, because they represent different motion and texture characteristics. The GOP structure is IPPP with 30 frames, the beginning 90 frames of each sequence are used for simulation unless otherwise noted. The reference frame number is one, in other words, only the previous frame is used for prediction. One slice is transmitted in one packet, taking the MTU of networks into account, we set the target slice length as 400 bytes [36] unless otherwise noted. Since at high bitrate, the slice number of one GOP could be large, 10-bit per RS symbol is used, namely Galois Field of $GF(2^{10})$.

[4]Matlab code of this work is available for download at http://www.mmtlab.com

So the value of N for the RS code (N, K) could be up to 1024, and the value of K depends on N and the per frame parity packet number evaluated using (4) in Section.III. We use the average luminance Peak Signal-to-Noise Ratio (PSNR) to assess the objective video quality, which is denoted as PSNR($\overline{mse}$), this is obtained by evaluating the Mean Squared Error (mse) over all the frames and over 200 trials, then the value of PSNR($\overline{mse}$) is calculated based on the averaged mse. It is worth mentioning that in all the following reported results, the bitrate includes both the video and parity packets. In our previous work [25], it was shown that the performance of Dynamic Sub-GOP FEC Coding (DSGF) approach is higher than many state-of-the-art approaches. Therefore, to have fair comparison we compare our results with DSGF [25] and Evenly FEC, both of which meet the real-time constraint and cause no additional delay.

In the first set of simulations, we study the effects of allocating different redundant packet rates for RS code. The network packet loss is i.i.d random packet loss model; for the same average packet loss rate $p = 10\%$, we try different RS redundant packet rates, μ, including $\{0.3, 0.4, 0.5, 0.6\}$. We do simulations with various quantization parameters (QP) to span a considerable bitrate range. Figure 3 shows the average PSNR versus bitrate curves with different RS redundant packet rates μ. In general, the PSNR curve for redundant packet rate 0.3 is much lower than other cases. The PSNR curves for $\mu = \{0.4, 0.5\}$ are very close; while in low bitrate, higher redundant rate, $\mu = 0.5$, can provide slightly better performance than that of $\mu = 0.4$, and vice versa, in high bitrate, lower redundant rate, $\mu = 0.4$, is slightly better. This is because in low bitrate, the slice number in each frame is small, which makes the performance of RS code low, and high RS redundant packet rate is required to compensate for this. For the PSNR curve of $\mu = 0.6$, although at low bitrate its performance is similar as that of $\mu = \{0.4, 0.5\}$, it is less performing in high bitrate. It it worth indicating that for a fixed total bitrate, higher redundancy means less bitrate could be used for the video date and vice versa; that is why having too high redundant packet rate cannot provide the best performance. In general, the PSNR curves for redundant rate $\{0.3, 0.4\}$ are similar; consequently, in the following simulations, we use RS redundant packet rate $\mu = 0.4$ for the 10% packet loss rate. Using the same methods, it is found that for packet loss rate $p = \{5, 10, 15, 20\}\%$, the proper RS redundant packet rate is $\mu = \{0.2, 0.4, 0.55, 0.7\}$, which means that μ should increase almost linearly with p. Therefore, in later simulations, RS redundant packet rate $\mu = 4p$ will be used. The precise relationship between μ and p is left for future investigation.

Figure 4 and 5 compare the performance of the three approaches in term of PSNR versus bitrate and for i.i.d average packet loss rate of 5% and 10%, respectively. Moreover, the H.264/AVC error free case is reported with the same H.264/AVC parameters that we used in the other three approaches, and this serves to show the up-bound of the performance. Clearly, for all the video sequences, and in the whole bitrate range, the RE-RS scheme outperforms the other two approaches significantly. Specifically, for the Foreman sequence and 10% i.i.d average packet loss rate, the proposed RE-RS scheme could provide 1.5 dB and 3.0 dB average gain over the DSGF approach and the Evenly FEC approach, respectively.

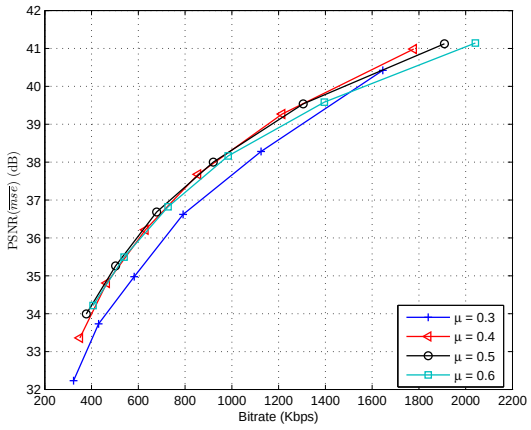


Figure 3. Average PSNR versus bitrate for various redundant packet rate μ; CIF Foreman sequence is used; i.i.d average packet loss rate is 10%; RS redundant packet rate μ includes $\{0.3, 0.4, 0.5, 0.6\}$.

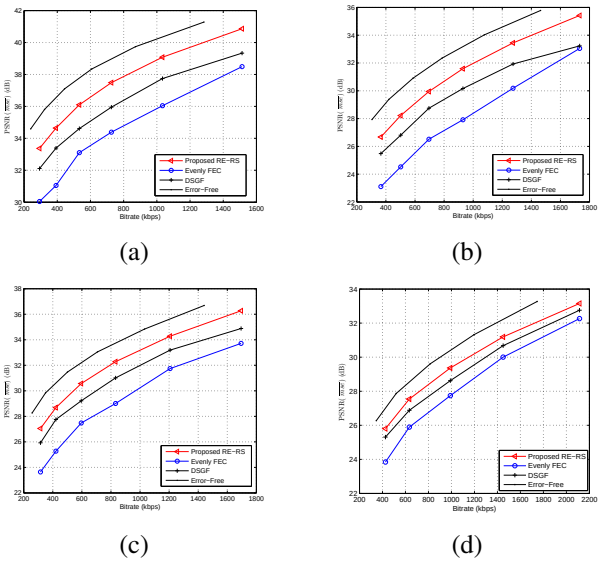


Figure 4. Average PSNR versus bitrate curves; i.i.d average packet loss rate is 5% and the redundant packet rate $\mu = 0.2$; (a) Foreman sequence, (b) Bus sequence, (c) Stefan sequence, (d) Mobile sequence.

To have a better understanding of the performance of the proposed RE-RS scheme, in Figure 6, its performance is compared with two ULP schemes [21], [37]. We select these two ULP schemes because both of them are implemented at frame-level, which means that they could be used for real-time applications, and this shares the same objective as the proposed RE-RS scheme. Meanwhile, these two ULP schemes are based on different criteria: [21] is based on the importance of each macroblock, so more protection is allocated for important macroblocks, whereas [37] is based on the concept of data partitioning. To have fair comparison, the same simulation setting as in [21] is used, where 300

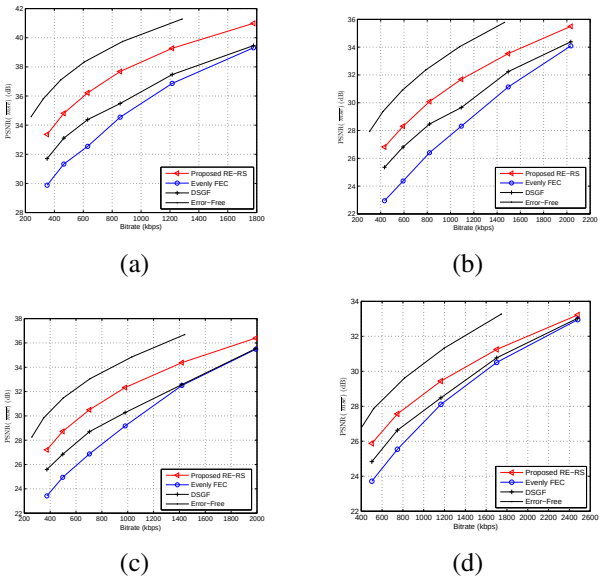


Figure 5. Average PSNR versus bitrate curves; i.i.d average packet loss rate is 10% and the redundant packet rate $\mu = 0.4$; (a) Foreman sequence, (b) Bus sequence, (c) Stefan sequence, (d) Mobile sequence.

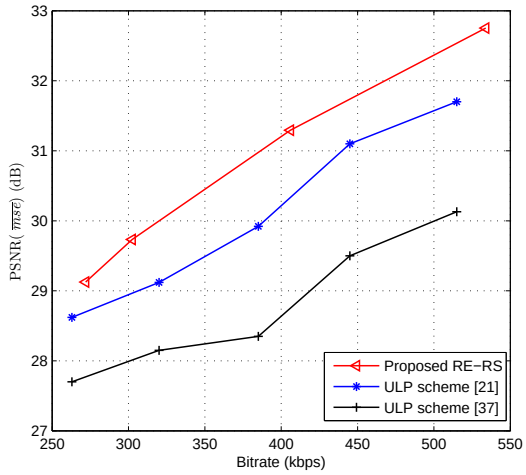


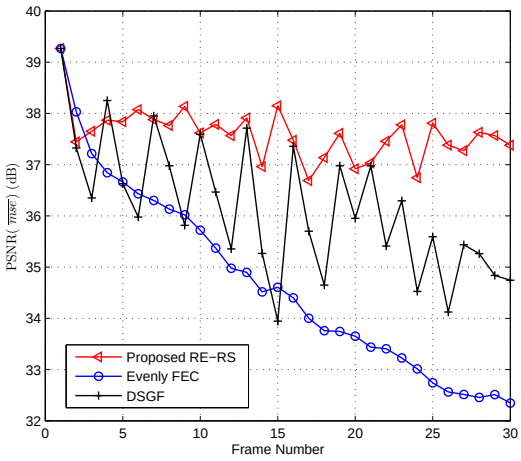
Figure 6. Performance comparison between the proposed RE-RS scheme and two ULP schemes [21], [37] ; CIF Paris sequence; i.i.d 10% average packet loss rate.

frames of the Paris sequence is used and the packet (slice) size is 200 byte. The performance curve of [37] is obtained from [21], which was used as the benchmark. As reported in Figure 6, the proposed RE-RS scheme outperforms both the two ULP schemes. The average gap between RE-RS and [21] is about 1 dB, whereas the performance improvement over [37] is even larger. Nevertheless, it should be mentioned that Paris sequence has low movement content, and typically, error concealment algorithm works well for this kind of video sequences. So it is expected that for the moderate and fast movement video sequences, the performance gain of RE-RS could be even larger.

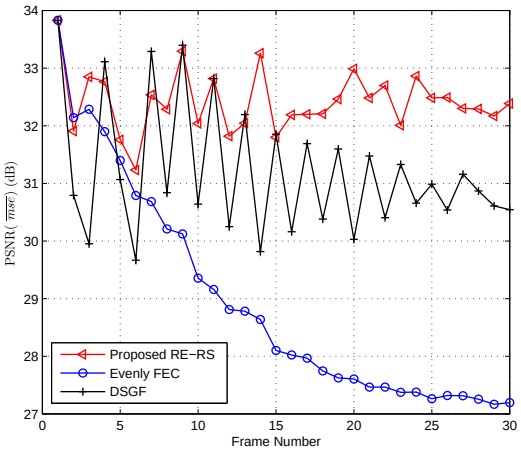
In Figure 7, with the Foreman and Stefan sequences, the frame by frame average PSNR curves, which are obtained by averaging the frame's mse of all the 200 trials and then evaluating the per frame PSNR, are plotted for the RE-RS scheme, Evenly FEC scheme and the DSGF approach. For the three approaches, the same QP is used to encode the video sequences, and the same amount of RS redundant packet rate is inserted to ensure fair comparison. It is shown that for almost all the frames, the average PSNR of the RE-RS scheme is higher than that of the Evenly FEC scheme. The gain increases with frame number, and at the last frame of the GOP, the average PSNR of the RE-RS scheme could be up to 5 dB higher than that of the Evenly FEC scheme. In the first half of the GOP, the peaks of the DSGF fluctuating PSNR could be as high as that of the RE-RS scheme, however, the PSNR of the RE-RS scheme is much less fluctuating. In fact, some PSNR bottoms of the Evenly FEC could be up to 4 dB lower than that of the RE-RS scheme. Moreover, in the second half of the GOP, even the PSNR peaks of DSGF approach fail to approach that of the RE-RS scheme. It is worth mentioning that similar results are obtained for the Bus sequence and for the other GOPs of the video sequences. In Figure 8, the average number of unrecovered packets among frames $[1, i]$, by the time of decoding frame i, is reported for Foreman sequence. From this figure it is observed that by the time of decoding frame i, the average number of unrecovered packets among frames $[1, i]$ for the proposed method is much smaller than the other two approaches for most of the frames. It is also noted that for all the three approaches, the RS code can recover all the lost packets in the first frame (I-frame), this is because the number of source packets in I-frame is large, and consequently the probability that the RS code fails is almost zero. This also explains why the average PSNR of the first frame in Figure 7 is higher than the other frames.

In all the previous experiments, i.i.d random packet loss model is used to simulate the network packet losses. In order to validate the performance of the proposed RE-RS in different error distribution models, in Figure 9 the PSNR versus bitrate curves in Gilbert burst loss model is reported. Since the error resilient performance of the DSGF approach [25] is much higher than the Evenly FEC approach, which was reported in [25], we compare the proposed RE-RS results with the DSGF approach. As indicated in [38], we set the average burst length as two. As it is expected, it is found that for both the RE-RS and DSGF approaches, the PSNR curves in burst loss environment are lower than that in i.i.d cases. It is also found that in burst loss cases, the average gain of the RE-RS scheme over the DSGF approach is 3.4 dB, being larger than that in i.i.d case, which is 1.5 dB. This is because in burst loss case, several consecutive packets tend to be lost together. In this case, with high probability, the RS code fails to recover the consecutively lost packets. Nevertheless, the burst packet losses are less catastrophic for the RE-RS scheme, this is because for the RE-RS scheme, if the RS code fails to recover the lost packets of the current frame, with high probability, they will be recovered by the expanding RS block of the following frames. Then the reference buffer will be updated, and error propagations will be stopped.

Figure 10 reports the PSNR versus bitrate curves for the low complexity sliding window schemes in both i.i.d and burst



(a) Foreman



(b) Stefan

Figure 7. Frame by Frame video quality in one GOP; i.i.d average packet loss rate is 5%, RS redundant packet rate $\mu = 0.2$. (a) Foreman Sequence; QP = 26. (b) Stefan Sequence; QP = 32.

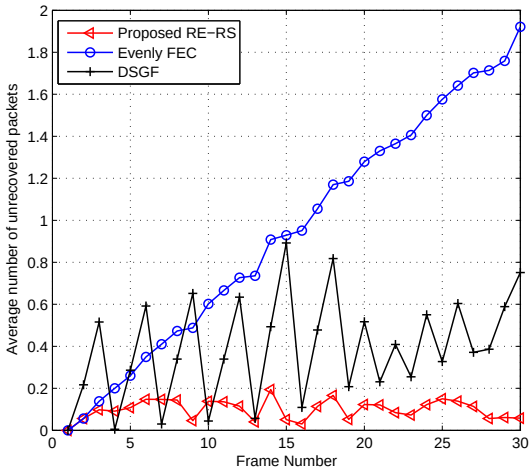


Figure 8. The average number of unrecovered packets among frames $[1, i]$ by the time of decoing frame i; i.i.d average packet loss rate is 5%, RS redundant packet rate $\mu = 0.2$; Foreman Sequence; QP = 26.

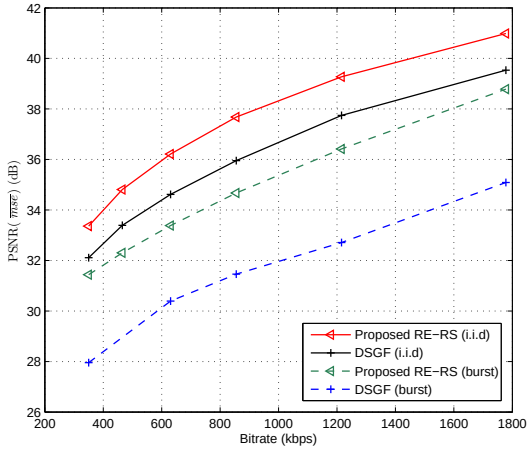
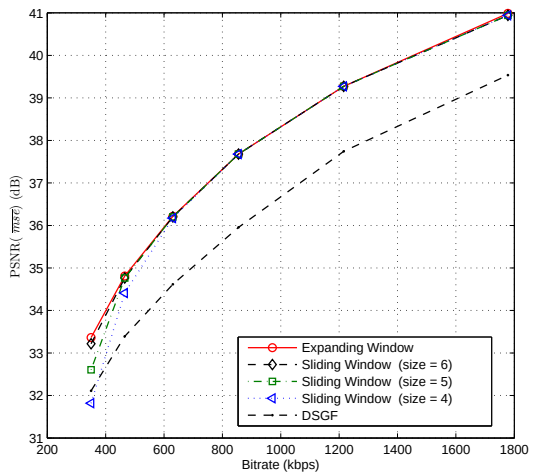


Figure 9. Average PSNR versus bitrate curves for the Foreman sequence; average packet loss rate is 10%, including i.i.d packet loss model and Gilbert burst packet loss model; for burst loss the average burst length is two.

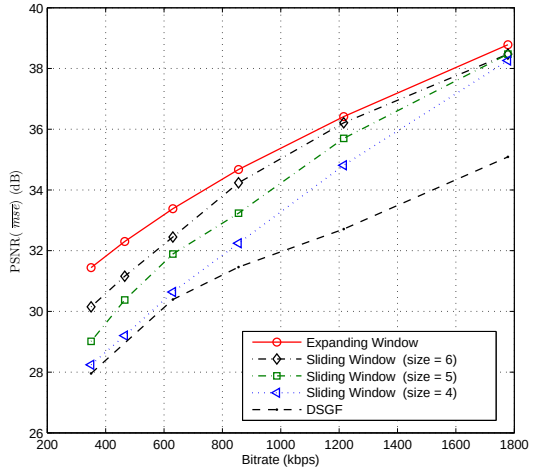
packet loss environments. It is noted that, for both i.i.d and burst packet loss models, the simplified sliding scheme outperforms the DSGF approach in all the bitrate range. Moreover, for the i.i.d case and in high and intermediate bitrate, the performance of sliding window scheme with window size of 4 is nearly the same as that of expanding window scheme, which suggests that for this simulation scenario RS window size of 4 frames are enough to recover most of the lost packets. The PSNR gap between the sliding window scheme and the expanding window scheme is larger in the burst loss case than in the i.i.d case, this is because burst packet losses are more difficult to recover, then it usually needs longer sliding window size. Meanwhile, in general, this gap is smaller in high bitrate than in low bitrate for both i.i.d and burst cases, because in high bitrate, the video packet number in each frame is large, which makes the RS code more efficient.

V. Conclusions

Facing one dilemma of traditional forward error correction coding of video streams, which is either low error correction performance or long FEC decoding delay, in this paper, a real-time error resilient video streaming scheme, named Randomized Expanding Reed-Solomon code, has been proposed. In this scheme, the RS coding block includes not only the video packets of the current video frame but also all the video packets of the previous frames in the current GOP. Thus, the error correction capability of the current frame could also be exploited to recover the lost packets of the previous frames. Therefore, the error propagations from the previous frames could be reduced significantly. To make the parity-check equations of the frames linearly independent, the randomly reordering technique has been proposed. Experimental results demonstrated that the proposed Expanding RS code scheme had considerable practical value for real-time video streaming applications.



(a) 10% i.i.d packet loss



(b) 10% burst packet loss

Figure 10. Average PSNR versus bitrate curves for expanding window and sliding window schemes; sliding window size 4, 5 and 6; Foreman sequence; (a) 10% i.i.d average packet loss rate, the redundant packet rate $\mu = 0.4$; (b) 10% burst packet loss, average burst length is 2, the redundant packet rate $\mu = 0.4$.

ACKNOWLEDGMENT

The authors would like to thank Associate Editor Prof. Eckehard Steinbach and the anonymous reviewers for their valuable comments and suggestions, which help to improve the paper significantly.

REFERENCES

[1] S. Wenger, "H.264/AVC over IP," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 645 – 656, july 2003.

[2] T. Stockhammer, M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 657 – 673, july 2003.

[3] W. Yao, S. Wenger, J. Wen, and A. Katsaggelos, "Error resilient video coding techniques," *IEEE Signal Processing Magazine*, vol. 17, no. 4, pp. 61 –82, Jul. 2000.

[4] R. Zhang, S. Regunathan, and K. Rose, "Video coding with optimal inter/intra-mode switching for packet loss resilience," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 966 –976, Jun. 2000.

[5] Y. Zhang, W. Gao, Y. Lu, Q. Huang, and D. Zhao, "Joint source-channel rate-distortion optimization for H.264 video coding over error-prone networks," *IEEE Transactions on Multimedia*, vol. 9, no. 3, pp. 445 –454, 2007.

[6] S. Wan and E. Izquierdo, "Rate-distortion optimized motion-compensated prediction for packet loss resilient video coding," *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1327 –1338, may 2007.

[7] H. Yang and K. Rose, "Optimizing motion compensated prediction for error resilient video coding," *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 108 –118, jan. 2010.

[8] J. Xiao, T. Tillo, C. Lin, and Y. Zhao, "Error-resilient video coding with end-to-end rate-distortion optimized at macroblock level," *EURASIP Journal on Applied Signal Processing*, vol. 2011, 2011:80.

[9] S. Soltani, K. Misra, and H. Radha, "Delay constraint error control protocol for real-time video communication," *IEEE Transactions on Multimedia*, vol. 11, no. 4, pp. 742 –751, 2009.

[10] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Transactions on Multimedia*, vol. 8, no. 2, pp. 390 – 404, april 2006.

[11] M. Schier and M. Welzl, "Optimizing selective ARQ for H.264 live streaming: A novel method for predicting loss-impact in real time," *IEEE Transactions on Multimedia*, vol. 14, no. 2, pp. 415 –430, april 2012.

[12] S. Lin, S. Mao, Y. Wang, and S. Panwar, "A reference picture selection scheme for video transmission over ad-hoc networks using multiple paths," in *IEEE International Conference on Multimedia and Expo, 2001. ICME 2001.* , 2001, pp. 96 – 99.

[13] W. Zia, K. Diepold, and T. Stockhammer, "Complexity constrained robust video transmission for hand-held devices," in *IEEE International Conference on Image Processing, 2007. ICIP 2007.* , vol. 4, 16 2007-oct. 19 2007, pp. IV –261 –IV –264.

[14] T. Tillo, M. Grangetto, and G. Olmo, "Redundant slice optimal allocation for H.264 multiple description coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 1, pp. 59 –70, 2008.

[15] C. Zhu, Y.-K. Wang, M. Hannuksela, and H. Li, "Error resilient video coding using redundant pictures," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 1, pp. 3 –14, 2009.

[16] I. Radulovic, P. Frossard, Y.-K. Wang, M. Hannuksela, and A. Hallapuro, "Multiple description video coding with H.264/AVC redundant pictures," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 1, pp. 144 –148, 2010.

[17] B. A. Heng, J. G. Apostolopoulos, and J. S. Lim, "End-to-end rate-distortion optimized md mode selection for multiple description video coding," *EURASIP Journal on Applied Signal Processing*, vol. 2006, no. 1, p. 12 pages, 2006.

[18] E. Baccaglini, T. Tillo, and G. Olmo, "Slice sorting for unequal loss protection of video streams," *IEEE Signal Processing Letters*, vol. 15, pp. 581 –584, 2008.

[19] A. Bouabdallah and J. Lacan, "Dependency-aware unequal erasure protection codes," *Journal of Zhejiang University-Science A*, vol. 7, pp. 27–33, 2006.

[20] X. Yang, C. Zhu, Z. G. Li, X. Lin, and N. Ling, "An unequal packet loss resilience scheme for video over the Internet," *IEEE Transactions on Multimedia*, vol. 7, no. 4, pp. 753 – 765, 2005.

[21] N. Thomos, S. Argyropoulos, N. Boulgouris, and M. Strintzis, "Robust transmission of H.264/AVC video using adaptive slice grouping and unequal error protection," in *2006 IEEE International Conference on Multimedia and Expo*, 2006, pp. 593 –596.

[22] H. Bobarshad, M. van der Schaar, and M. Shikh-Bahaei, "A low-complexity analytical modeling for cross-layer adaptive error protection in video over WLAN," *IEEE Transactions on Multimedia*, vol. 12, no. 5, pp. 427 –438, aug. 2010.

[23] L. Toni, P. Cosman, and L. Milstein, "Channel coding optimization based on slice visibility for transmission of compressed video over OFDM channels," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 7, pp. 1172 –1183, august 2012.

[24] J. Xiao, T. Tillo, C. Lin, and Y. Zhao, "Real-time forward error correction for video transmission," in *2011 IEEE Visual Communications and Image Processing (VCIP)*, nov. 2011, pp. 1 –4.

[25] ——, "Dynamic Sub-GOP forward error correction code for real-time video applications," *IEEE Transactions on Multimedia*, vol. 14, no. 4, pp. 1298 –1308, 2012.

[26] P. Cataldi, M. Grangetto, T. Tillo, E. Magli, and G. Olmo, "Sliding-window raptor codes for efficient scalable wireless video broadcasting with unequal loss protection," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1491 –1503, june 2010.

[27] D. Vukobratovic, V. Stankovic, D. Sejdinovic, L. Stankovic, and Z. Xiong, "Scalable video multicast using expanding window fountain codes," *IEEE Transactions on Multimedia*, vol. 11, no. 6, pp. 1094 – 1104, oct. 2009.

[28] C. Hellge, D. Gomez-Barquero, T. Schierl, and T. Wiegand, "Layer-aware forward error correction for mobile broadcast of layered media," *IEEE Transactions on Multimedia*, vol. 13, no. 3, pp. 551 –562, june 2011.

[29] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103 –1120, sept. 2007.

[30] M. Luby, A. Shokrollahi, M.Watson, T. Stockhammer, and L. Minder, RaptorQ Forward Error Correction Scheme for Object Delivery, IETF RMT draft-ietf-rmt-bb-fec-raptorq-04, Aug. 2010. [Online]. Available: http://tools.ietf.org/html/draft-ietf-rmt-bb-fec-raptorq-04.

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

[32] I. Rhee and S. Joshi, "Error recovery for interactive video transmission over the Internet," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 1033 –1049, jun 2000.

[33] M. Ghanbari, "Postprocessing of late cells for packet video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 6, pp. 669 –678, dec 1996.

[34] J. Brennan and J. Wolfskill, "Remarks on the probability the determinant of an n×n-matrix over a finite field vanishes," *Discrete mathematics*, vol. 67, no. 3, pp. 311–313, 1987.

[35] http://iphome.hhi.de/suehring/tml/download/.

[36] P. Baccichet, R. Shantanu, and G. Bernd, "Systematic lossy error protection based on h. 264/avc redundant slices and flexible macroblock ordering," *Journal of Zhejiang University-Science A*, vol. 7, no. 5, pp. 900–909, 2006.

[37] O. Harmanci and A. Tekalp, "Stochastic frame buffers for rate distortion optimized loss resilient video communications," in *IEEE International Conference on Image Processing, 2005. ICIP 2005.*, vol. 1, sept. 2005.

[38] D. Loguinov and H. Radha, "End-to-end internet video traffic dynamics: statistical study and analysis," in *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2002, pp. 723 – 732 vol.2.

Jimin XIAO was born in Suzhou, China. He received the BS and MEng degrees in telecommunication engineering from Nanjing University of Posts and Telecommunications, China, in 2004 and 2007, respectively. Then he worked as a software engineer in Motorola (China) Electronics Ltd, and later as system engineer in Realsil (Realtek) Semiconductor Corp. Currently, he is persuing his PhD degree in the University of Liverpool, UK. His research interests are in the areas of video streaming, image and video compression, and multiview video coding.

Tammam TILLO (M05-SM12) was born in Damascus, Syria. He received the Engineer Diploma in Electronic Engineering from Damascus University, Damascus, Syria, in 1994, and the Ph.D. in Electronics and Communication Engineering from Politecnico di Torino, Torino, Italy, in 2005. From 1999 to 2002 he was with Souccar for Electronic Industries, Damascus, Syria. In 2004 he was visiting researcher at the EPFL (Lausanne, Switzerland), and from 2005 to 2008, he worked as a Post-Doctoral researcher at the Image Processing Lab of Politecnico di Torino, and for few months he was Invited Research Professor at the Digital Media Lab, SungKyunKwan University, Suwon, S. Korea. In 2008 he joined Xian Jiaotong- Liverpool University (XJTLU), Suzhou, China. Currently, he is the Head of Electrical and Electronic Engineering Department and Acting Head of Department, Department of Computer Science and Software Engineering at XJTLU university. His research interests are in the areas of robust image and video transmission, image and video compression, and hyperspectral image compression.

Yao ZHAO (M'06-SM'12) received the B.S degree from Fuzhou University in 1989 and the M.E degree from the Southeast University in 1992, both from the Radio Engineering Department, and the PhD degree from the Institute of Information Science, Beijing Jiaotong University (BJTU) in 1996. He became an associate professor at BJTU in 1998 and became a professor in 2001. From 2001 to 2002, he worked as a senior research fellow in the Information and Communication Theory Group, Faculty of Information Technology and Systems, Delft University of Technology, Netherlands. He is now the director of the Institute of Information Science, Beijing Jiaotong University. His research interests include image ideo coding, fractals, digital watermarking, and content based image retrieval. Now he is leading several national research projects from 973 Program, 863 Program, the National Science Foundation of China. He was the recipient of the National Outstanding Young Investigator Award of China in 2010.