A Real-Time Error Resilient Video Streaming Scheme Exploiting the Late- and Early-Arrival Packets

Jimin Xiao, Tammam Tillo, *Senior Member, IEEE,* Chunyu Lin, Yungang Zhang, and Yao Zhao, *Senior Member, IEEE*

***Abstract*—For real-time video streaming systems, the video packets arriving after the display deadline of their frames are considered as *late-arrival* packets, and typically they are discarded. This will affect the current frame and the following ones due to error propagations. For this reason, in this paper, we propose an approach to exploit the *late-arrival* and out-of-order packets, which includes two mechanisms. The first mechanism will use these packets to update the reference frames to make them more consistent with the encoder side, and this will eventually reduce the error propagations. The second mechanism will use these packets to increase the chance of successfully decoding the Reed-Solomon (RS) code. In the proposed approach, a sub-GOP based systematic RS code is used and optimized to exploit these packets, where the size of each sub-GOP and the parity packet number for each sub-GOP are optimally tuned, taking into consideration the maximum end-to-end delay, the network conditions, and other system parameters, so as to make the best use of the *late-arrival* packets and to exploit the out-of-order packets. Finally, the experimental results show the advantage of the proposed approach over other approaches.**

***Index Terms*—late-arrival packet, early-arrival packet, sub-GOP, error propagation, reference buffer, error resilience, H.264/AVC**

I. Introduction

LOW delay video steaming in unreliable network environments, i.e., wireless network or packet-switched network, is a challenging task. 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 in video coding, video transmission, and the nature of errors and losses, is an important task.

When the video packets are transmitted over lossy networks, some packets are dropped by the underlying network facilities, and they never reach the destinations, thus we call them *physical lost* packets. Moreover, some packets may arrive at the destination after long delay, which is larger than the maximum allowed end-to-end delay for the applications. In general, these packets are also regarded as lost packets by the video applications. To distinguish these packets from the *physical lost* packets, in the following, we will refer to them as *late-arrival* packets. Meanwhile, for some video packets, the transmission delay is short, and they may arrive at the destination before the display deadline of their temporal-previous frames, in this article those are called *early-arrival* packets. It is worth recalling that the end-to-end delay constraint is an application-dependent parameter, so for example, for real-time video conferencing/telephony applications, the acceptable end-to-end delay is between 150 ms and 400 ms according to ITU-T G.114 [3].

In literature, many error resilient techniques have been proposed, overviews of error resilient techniques were presented in [4], [5]. These methods mainly aim to mitigate the effects of physical lost packets. One commonly used approach for this is to insert some intra macroblocks to cut off the propagation paths of the distortion caused by the losses. In [6], [7], the end-to-end distortion, which includes both the source coding errors and the channel-induced errors, is estimated at the encoder side, then the intra macroblocks are optimally inserted based on the end-to-end R-D optimization. However, the coding efficiency of the intra macroblock refreshment approaches is compromised, since the coding efficiency of intra mode is typically several times lower than inter mode. On the other hand, the efficient methods based on Automatic Repeat reQuest (ARQ) [8], [9] and feedback-based Reference Picture Selection [10], which also aim to stop the propagated errors, cause one round-trip time (RTT) delay, this makes them unsuitable for real-time video applications. Whereas, redundant slice/picture coding with equal or lower quality

Manuscript received September 26, 2012; revised April 5, 2013; accepted April 8, 2013. Date of publication June 21, 2013; date of current version August 21, 2013. This work was supported in part by the National Natural Science Foundation of China under Grants 60972085 and 61210006, the National Basic Research Program of China under Grant 2012CB316400, the National Science Foundation of China under Grant 61025013. *(Corresponding author: T. Tillo.)*

J. Xiao is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3GJ, U.K., and also with the Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China (e-mail: jimin.xiao@liverpool.ac.uk).

T. Tillo is with the Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China (e-mail: tammam.tillo@xjtlu.edu.cn).

C. Lin and Y. Zhao are with the Institute of Information Science, Beijing Jiaotong University, Beijing 100044, China, and also with the Beijing Key Laboratory of Advanced Information Science and Network Technology, Beijing 100044, China (e-mail: cylin@bjtu.edu.cn; yzhao@bjtu.edu.cn).

Y. Zhang is with the Department of Computer Science, University of Liverpool, Liverpool L69 3GJ, U.K., and also with the Department of Computer Science, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China (e-mail: yungang.zhang@liverpool.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBC.2013.2258111

[11], [12] and multiple description coding (MDC) [13]–[15] usually causes no additional delay. However, in the redundant slice/picture based approaches, when the redundant version is used to replace the primary one, mismatch errors will appear in the decoding loop, and in general they will propagate to the following frames until the end of the Group of Pictures (GOP). One more set of effective approaches to combat packet losses is based on the use of Forward Error Correction (FEC) codes, it has been widely used in the video broadcasting system, such as DVB-H system [16], [17]. However, most FEC based approaches introduce a considerable delay. For example, in the FEC scheme applied at GOP level [18], [19], one GOP of delay is caused. Nevertheless, the error resilient performance of FEC scheme in [18], [19] has been shown to outperform Redundant Slice MDC (RS-MDC) [11]. In [20], the FEC coding block has been made to contain frames from one block of packets (BOP), where different packets are unequally protected against losses based on both the frame position and the packet type, as defined by data partitioning paradigm. So for this approach, which is based on MPEG-4, one BOP of delay is caused, and the delay depends on the length of the BOP. Moreover, because the packets from one BOP are divided into two FEC coding blocks, based on their data partitioning type, the performance of the FEC code is compromised. In [21]–[23], FEC and retransmission are jointly used, and this approach is only suitable for the application that has a relatively loose end-to-end delay constraint, i.e, allowing delay larger than the RTT.

In this paper, a Real-time Video Streaming scheme exploiting the Late- and Early-arrival packets (RVS-LE) in an optimal fashion is proposed. In the proposed approach, we are targeting real-time applications with stringent end-to-end delay constraint, e.g., delay less than one RTT, thus retransmission is not considered. The proposed RVS-LE approach includes two mechanisms. One is to use these packets to update the reference frames to make them more consistent with the encoder side, and this will eventually reduce the error propagations. The other mechanism is to use these packets to increase the chance of successfully decoding the RS code. It is worth mentioning that exploiting the *late-arrival* packets to update the reference and stop error propagations was also used in [24], [25]. Whereas in this paper, the *late-arrival* packets will be jointly optimized with the FEC code in an optimal way. To further exploit the *late-* and *early-arrival* packets, sub-GOP based systematic Reed-Solomon (RS) code [26], [27], which is proposed to improve the error correction performance of the RS code, is also used to increase the probability of successfully decoding the current frame. Finally, in the proposed RVS-LE approach, the size and the parity packet number for each sub-GOP are optimally tuned, taking into consideration the maximum end-to-end delay, the network conditions, which accounts for the *physical lost* packets, *late-* and *early-arrival* packets to minimize the total distortion, under the constraints of the inserted redundancy and the maximum end-to-end delay. It is worth noticing that in our preliminary work [26], [27], the network transmission delay, the *late-arrival* packets were not taken into consideration during the sub-GOP and parity allocation process, which makes them less optimal for practical video streaming applications, since in practical applications, the video packets take half RTT delay to reach the destinations.

The rest of the paper is organized as follows. A brief review of the systematic RS erasure code is provided in Section II. In Section III, the proposed RVS-LE approach is presented. Later, the frame level Evenly FEC approach is introduced, this approach is used as a benchmark for the proposed RVS-LE coding. In Section IV some simulation results validating the proposed approach are given. Finally, some conclusions are drawn in Section V.

II. Systematic Reed-Solomon Erasure Code

The most common implementation of FEC is using RS code [28], [29]. The systematic RS erasure code has been widely used as FEC code to protect data packets against losses in packet erasure networks. In this section, we will briefly recall some concepts and notations about systematic RS erasure code, which will be used throughout this paper. In systematic RS (N, K) code, for every K source packets, $(N - K)$ parity packets are introduced to make up a codeword of packets. As long as a client receives at least K out of the N packets (or in percentage K/N, which is defined as the RS code rate), 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.

One important parameter about the RS code, that we will need is the remaining packet loss rate after the RS correction (p'), this could be evaluated as

$$p' = \frac{\sum_{i=1}^{K} i \ p_{rs}(i)}{K} \tag{1}$$

with $p_{rs}(i)$ representing the probability of still having i unrecoverable source packets after RS correction. From now on we will refer to those packets as unrecoverable lost packets. To evaluate $p_{rs}(i)$, let us use p to denote the network packet loss rate, use $p_s(n)$ and $p_r(n)$ to denote the probability of losing n packets before decoding the RS code among the source packets and parity packets, respectively.

$$p_s(n) = \binom{K}{n} (1 - p)^{K-n} p^n \tag{2}$$

$$p_r(n) = \binom{N-K}{n} (1 - p)^{N-K-n} p^n \tag{3}$$

Since having i unrecoverable lost packets is caused by losing i source packets, and at the same time losing more than $N-K-i$ RS parity packets. So the probability of having i unrecoverable lost packets is

$$p_{rs}(i) = \begin{cases} p_s(i) \ P_r(N - K - i + 1) & i \le N - K \\ p_s(i) & i > N - K \end{cases} \tag{4}$$

where $P_r(j)$ is used to denote that no less than j RS parity packets are lost. Based on (3), the value of $P_r(j)$ is

$$P_r(j) = \sum_{n=j}^{N-K} p_r(n) \tag{5}$$

During designing the systematic RS erasure code, one interesting property related to its correction capability needs to be taken into account. To review this property, let μ be the rate of the inserted parity packets, *i.e.*, $\mu = (N - K)/K$. In this case, the total number of transmitted packets is $N = K + \mu K$. To be able to recover pN losses, we need to have $\mu K \geq pN$ or equivalently $\mu \geq \mu_{min} = \frac{p}{1-p}$. In the limit case, according to the law of large number, when $N \rightarrow \infty$, then μK could be as small as pN, namely $\mu K \approx pN$, *i.e.*, $\mu = \mu_{min}$. This means for the same parity packet rate, the larger the value of K is, the higher the performance of RS code can be.

III. Proposed Video Streaming Approach

Since our objective is to design a video steaming system for real-time delay constraint applications, with a maximum allowed end-to-end delay, the B-frames will not be used in order to minimize the delay in the video encoding process, so we will use the IPPP GOP structure. 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, 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. The packet encapsulation process is demonstrated in Fig.1, where for those 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.

A. RVS-LE Approach: A Case Study

In the proposed approach, all the video packets arriving at the destination with different delay are exploited, including *late-* and *early-arrival* packets. To do this, we propose to incorporate packets from a Sub-GOP, which usually contains more than one video frame, to one RS coding block, and add parity packets at the end of the sub-GOP. In this case, both *late-* and *early-arrival* packets could be used by the RS code to recover the still unavailable packets.

Fig.2.a shows one example of how to exploit the *late-* and *early-arrival* packets, where slices of one sub-GOP (3 frames) are used as one RS coding block, and the RS parity packets are allocated and appended at the end of each sub-GOP, thus RS code (15, 12) is used for this sub-GOP. Here packets 3 and 7 are *late-arrival* packets, because they arrive at the destination later than the display deadlines of the frames they belong to; whereas, packets 6, 10, 11, 12, 13 and 14 are *early-arrival*

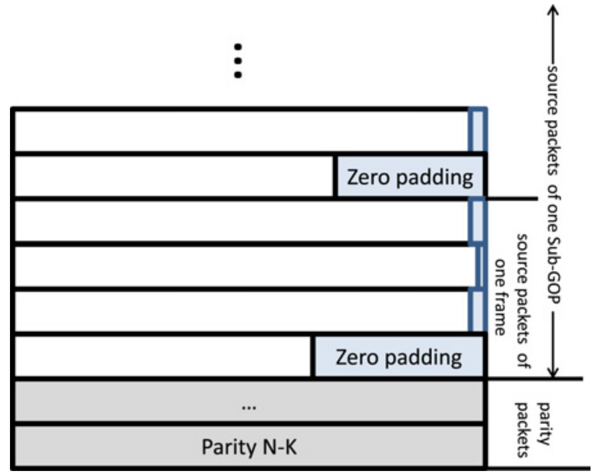

Fig. 1. Video slices encapsulation and parity packets appending, H.264/AVC fixed slice length in term of byte is used, the video slice length is nearly the same except for the last one in each frame.

packets, as they arrive at the destination before the display deadlines of their temporal-previous frames.

The decoder will decode the first frame with slice 3 being concealed. However, this packet arrives at the destination by the display deadline of the second frame, so in order to stop the error propagations caused by the injected errors in the prediction loop by the concealment stage, the first frame will be re-decoded with the newly received slice. It is worth indicating that the updated version of the first frame will not be displayed, nevertheless, it will be used to update the reference buffer, so as to stop the propagated distortion. By the display deadline of frame 2, slice 7, which belongs to frame 2, does not arrive at the destination. However, with four early-arrival packets, *i.e.*, 10, 12, 13, 14, in addition to the other arrived packets belonging to the current sub-GOP, which totally amounts to twelve packets, the RS code will be able to recover packet 7. This allows to correctly decode frame 2 and recover packet 9, *i.e.*, the lost packet in frame 3. In this case, for frame 2 and 3, there is no concealment distortion propagated from the first frame, and no distortion will propagate to the following frames.

Fig.2.b is another example that serves to demonstrate how the *late-* and *early-arrival* packets are exploited between different sub-GOPs. In this example, by the display deadline of frame 3, the amount of received packets for the first sub-GOP is not enough to recover all its source packets. Thus, there will be concealment distortion for the lost packets 3, 9. Whereas for the *later-arrival* packet 7, it will arrive at the destination before the display deadline of frame 3. Thus before decoding frame 3, this slice will be re-decoded and updated. It is worth noticing that the *early-arrival* packet 17 belongs to the second sub-GOP, and it cannot be used by the RS code of the first sub-GOP. Later, packet 13 arrives before the deadline of the fourth frame ($t4$), and with this newly received packet, the total received packets for the first sub-GOP becomes twelve, which allows the RS-decoder to recover all the lost packets of the first sub-GOP, and consequently, the video decoder will re-decode all the video frames of the first sub-GOP

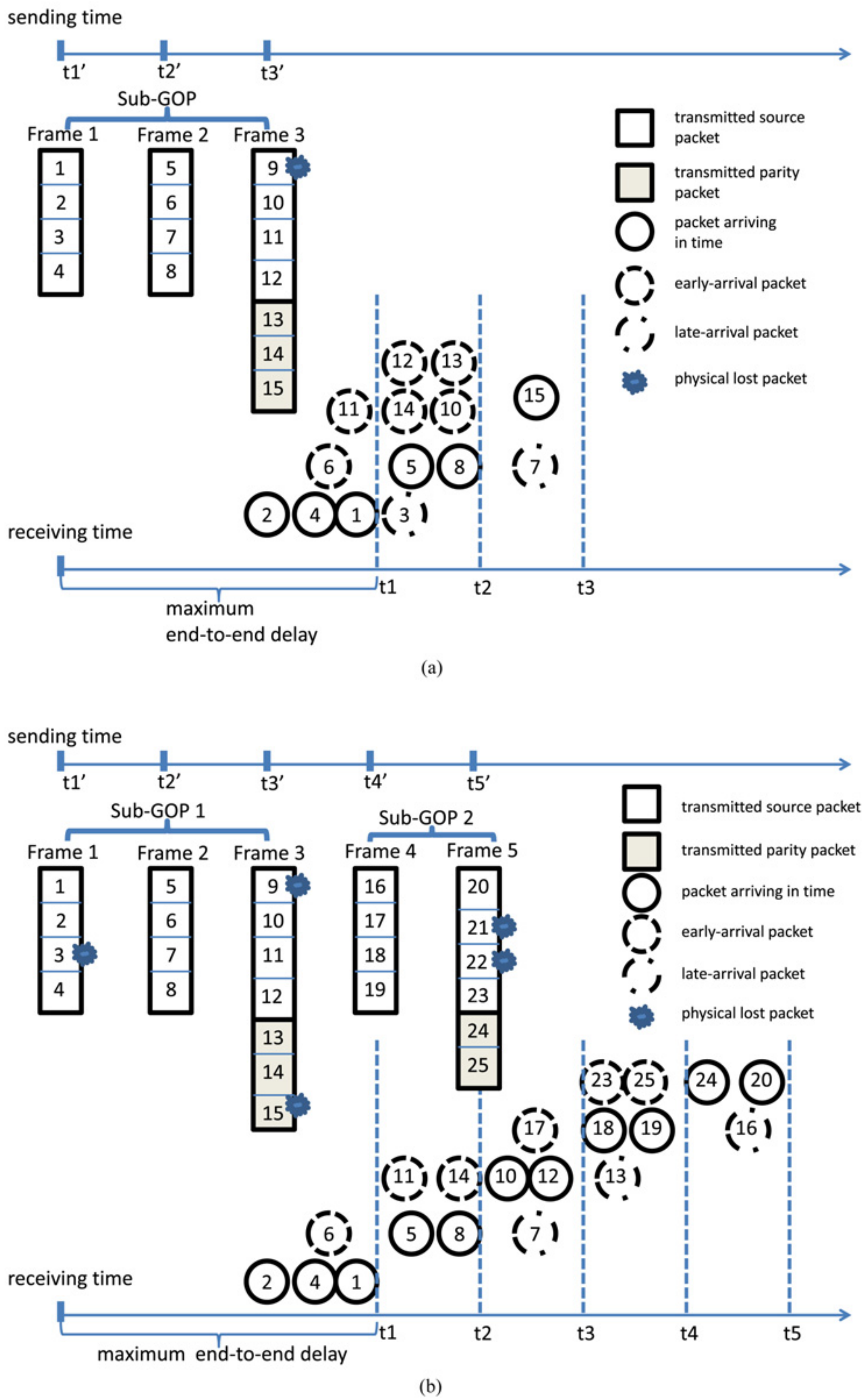

Fig. 2. Examples of sub-GOP and RS parity packets allocation and packet transmission delay for the proposed approach; the I-frame (with index 0) is not shown in the figure; $t1$, $t2$, $t3$, $t4$, $t5$ are the display deadline for frame 1, frame 2, frame 3, frame 4 and frame 5, respectively; $t1'$, $t2'$, $t3'$, $t4'$, $t5'$ are the sending time for frame 1, frame 2, frame 3, frame 4 and frame 5, respectively. (a) Example with one sub-GOP. (b) Example with two sub-GOPs.

using the previously arrived and RS-recovered packets, and following that the reference buffer will be updated. Thus, the concealment distortion for losing slice 3 and 9 will stop propagating to the following frames, which means that the mismatch caused by the concealed slice 3 may only affect frames 2 and 3, whereas the concealed slice 9 only affects frame 3.

As demonstrated in the above examples, the general principal of the proposed RVS-LE approach could be described as follows. On the receiver side, all the received packets of the current frame-to-be-displayed will be decoded and displayed at the display deadline. If some packets are not available then the RS-decoder will try to recover these missing packets using the already received packets. Note that the already received packets may include the *late-arrival* packets of the temporal-previous frames, the *early-arrival* packets of the following frames and the received parity packets of the current sub-GOP. Finally, for all the cases where the RS-decoder fails to recover the missed packets of the current frame, the error concealment will be invoked. It is also important to note that in some cases, at the end of the current sub-GOP the total number of received packets may not be enough to make the RS decoder recover all the missed packets, this sub-GOP will be described in the following as *pending* sub-GOP, in other words, the decoding process of these sub-GOPs will be regarded as uncompleted. Moreover, the *pending* sub-GOP category will also include those sub-GOPs that are completely decoded (all their packets arrived or recovered), however, their previous sub-GOPs are *pending*, because in this case the mismatch error may propagate from the previous sub-GOPs to the following ones. In order to mitigate the mismatch error, each newly received packet will be checked to see whether it belongs to any previously *pending* sub-GOPs. If one or more newly arriving packets belong to a *pending* sub-GOP, then they will be firstly used together with the previously received packets that belong to this *pending* sub-GOP, by the RS-decoder, to try to recover its missing packets. Secondly, with the recovered missing packets and the newly received source packets, the *pending* sub-GOP will be re-decoded by the video-decoder. If the previously described stage leads to an update of the content of the previous frames then the reference buffer frame will be updated and also all the following sub-GOPs will be re-decoded and updated.

B. Optimal sub-GOP and RS Parity Packet Allocation

This section will address the problem of how to group frames into sub-GOPs, and how to allocate the RS parity packets among all the sub-GOPs in order to maximize the exploitation of the *late-* and *early-arrival* packets. It is worth noticing that, in the proposed approach, I-frames are not included in any sub-GOP, and are protected at frame level. This is because, in general, I-frames generate much more bits than P-frames, and therefore more source packets are produced for I-frames, and consequently, protecting them directly is efficient enough. While for the allocation problem of the P-frames, we have to note that, on one hand, if the sub-GOP includes too few P-frames, the *late-* and *early-arrival* packets cannot be exploited properly, and the value of K for the RS code will not be large enough to make the RS code efficient. On the other hand, if the sub-GOP includes too many P-frames, the time interval for the whole sub-GOP will be very long. Having a long time interval means that by the display deadline of many frames of the current sub-GOP, the amount of received packets of this sub-GOP will be, with high probability, too few to allow the RS-decoder to recover any unavailable packets. To optimally generate the sub-GOPs and properly allocate them the RS parity packets, we need to know some detailed information of this GOP, including the number of P-frames in one GOP, the slice number in each frame, the concealment distortion caused by losing each slice, and how the distortion propagates. However, the actual values of some of these parameters are not available for real-time on-the-fly transmission system. To overcome this limitation, we use the expected values of these parameters instead of their actual values, which will obviously make the obtained solution sub-optimal. Nevertheless, the obtained results demonstrate the effectiveness of the proposed procedure. These parameters are listed in the following.

1) Instead of the actual number of slices in each P-frame, for each GOP, we will use the average number of slices per P-frame, $\overline{S}$. Therefore, to predict this value we will assume that it does not change from GOP to GOP, and consequently, for a practical approach we will assume that the current $\overline{S}$ is similar to the actual value of the average number of slices per frame of the previous GOP, which could be obtained at the end of the encoding process of the previous GOP. This approach becomes more accurate for low motion video with little change in the content.
2) The actual concealment distortion caused by losing a slice will be replaced by the expected concealment distortion of losing one slice, $\bar{d}$, we assume $\bar{d}$ to be the same for all slices. To estimate the amount of propagated distortion to the following frames, an attenuation function $f(n)$ will be used [30], [31]. This means that if the concealment distortion of one slice is $\bar{d}$, it will propagate to the following frames and become $f(n)\bar{d}$ after n frames. For the sake of simplicity the function $f(n) = \alpha^{n-1}$ $(0 < \alpha \leq 1)$ is employed, this expression approximates, at low levels of attenuation, the function $f(n) = (1 + \lambda_1 n)^{-1}$ and $f(n) = (1 + \lambda_2 n)^{-1/2}$ reported in [30] and [31], respectively, with α, λ_1 and λ_2 being parameters that depends on the sequence itself. Moreover, in order to model multiple slice losses, let us assume that the distortion caused by losing multiple slices are uncorrelated, in this case, the total expected distortion for the whole GOP will be the sum of the expected distortions caused by losing the individual slices. The assumption that slice concealment distortion is uncorrelated is reasonable. In fact, concealment distortions, can be considered as uncorrelated with the pixel values, then concealment distortions caused by losing different slices can also be considered as uncorrelated. The additive distortion model has been verified experimentally in [11].

One more parameter that we need to take into account, while solving the allocation problem, is the maximum allowed end-to-end delay (T_{max}). In fact, this parameter will determine the probability of receiving packets at the display deadline of each frame. Let us use $p_{k,i}$ to denote the probability of receiving packets belonging to frame i at the display deadline of frame k, this probability could be evaluated as:

$$p_{k,i} = cdf\,(T_{max} + (k-i)\ \ T_0) \tag{6}$$

with T_0 being the time interval between two adjacent frames, e.g., for 30 fps application T_0 would be 33.3 ms; and $cdf(t)$ is the Cumulative Distribution Function (CDF) of the packet delay, which means the ratio of packets with delay less than t is $cdf(t)$, this function is dictated by the network conditions. In (6) the frame index i could be either larger or smaller than k, with the condition that $T_{max}+(k-i)\ T_0 > 0$, so for example by the display deadline of frame k the probability of receiving packets belonging to the previous frame is $cdf(T_{max}+T_0)$, this is because the total waiting time for the packets belonging to frame $k-1$ is $T_{max}+T_0$. Moreover, to solve the allocation problem, the maximum number of parity packets for all the P-frame in one GOP, R, need to be specified, and consequently the number of RS parity packets allocated for the ith P-frame, $R(i)$, need to satisfy the following constraint:

$$\sum_{i=1}^{L} R(i) \le R \tag{7}$$

with L being the number of P-frames in one GOP; it is worth noting that if for example $R(1) = R(2) = 0$ and $R(3) = 3$ then that means that the first, second, and third frames form one sub-GOP protected by three RS parity packets, this case describes the first sub-GOP in Fig.2.b and the sub-GOP in Fig. 2.a.

At this stage, with all the parameters and the constraints listed above, we could tune the sub-GOP size and allocate the parity packets. To solve this problem, we assume that these packets will be inserted in totally t positions, these positions are identified by the frame indexes $r_1,\ r_2,\dots,\ r_t$, whereas for all the other P-frames no RS parity packets will be inserted. Moreover, let us assume that the number of the inserted RS parity packets is $R(r_1), R(r_2),\dots,\ R(r_t)$. According to this notation, the RS parity packets allocated for frame r_{m+1} are used to protect frames $[r_m+1,\ \ r_{m+1}]$, and this sub-GOP will be indexed as the $(m+1)\ th$ sub-GOP [1]. Therefore, the RS code used for this sub-GOP would be $(N,K) = \big((r_{m+1}-r_m)\ \overline{S} + R(r_{m+1}), (r_{m+1}-r_m)\ \overline{S}\big)$. For the example reported in Fig.2.b, the allocated RS parity packets are in the following two positions $r_1 = 3,\ r_2 = 5$, *i.e.*, $t = 2$, and $R(r_1) = 3,\ R(r_2) = 2$.

Now let us evaluate the expected distortion in frame k, with $k \ge r_m+1$, caused by slice losses within sub-GOP $m+1$, *i.e.*, frames $[r_m+1,\ \ r_{m+1}]$. To evaluate this expected distortion, let us denote the number of source packets sent at time i and have not arrived at the receiver side at the display deadline of frame k by $u_{k,i}$, and the probability of this event by $P(u_{k,i})$. Similarly, $v_{k,r_{m+1}}$ will be used to denote the number of RS parity packets sent at time r_{m+1} and have not arrived by the display deadline of frame k, and the probability of this event is $P(v_{k,r_{m+1}})$. As previously described, the probability of receiving a packet belonging to the i-th frame, by the display deadline of the k-frame, is $p_{k,i}$, therefore, the probability $P(u_{k,i})$ becomes

$$P(u_{k,i}) = \binom{\overline{S}}{u_{k,i}} (p_{k,i})^{\overline{S}-u_{k,i}} (1-p_{k,i})^{u_{k,i}} \tag{8}$$

Similarly, the probability of the event $v_{k,r_{m+1}}$ is shown in (9). According to the property of systematic RS code, the distortion in frame k caused by the unavailable slices within sub-GOP $m+1$, is because the total number of received packet of this sub-GOP is less than K by the display deadline of frame k. In other words, the RS decoder cannot recover the unavailable source packets within this sub-GOP. Thus, in order to evaluate the expected distortion in frame k, let us define the set $\mathcal{I}_{k,m+1}(z)$ as in (10). This set stands for all the possible patterns of packet reception, that have totally z unavailable packets in the $(m+1)th$ sub-GOP by the display deadline of frame k. Therefore, the expected distortion in frame k ($k \ge r_m+1$) caused by the $(m+1)th$ sub-GOP could be formulated as (11). As previously defined, $f(n)$ is the distortion attenuation function, $\overline{d}$ is the expected distortion caused by each slice. It is important to note that, the term $\bar{d}_{m+1}(k)$ contains all the distortion in frame k that is caused by the $(m+1)th$ sub-GOP, including the propagated distortion from the previous frames of this sub-GOP. So (11) is obtained by adding up the weighted concealment and propagated distortion of all the packet loss patterns that make the RS code fail to recover the unavailable packets, where the weight factor is the probability of all the specific patterns. Consequently, the total expected distortion caused the by $(m+1)\ th$ sub-GOP could be obtained by summing up the caused distortion within frames $[r_m+1, L]$ as

$$\bar{D}_{m+1} = \sum_{k=r_m+1}^{L} \bar{d}_{m+1}(k) \tag{12}$$

with L being the number of P-frames in one GOP. Finally, by adding up the expected distortion of all the sub-GOPs within the GOP, the total expected distortion of the whole GOP is

$$\bar{D}_{total} = \sum_{k=1}^{r_t} \bar{D}_k \tag{13}$$

At this stage, the optimization problem can be formulated as the following constrained minimization:

$$\begin{cases} \text{min} & \bar{D}_{total} \\ \text{subject to} & \sum_{i=1}^{L} R(i) \le R \end{cases} \tag{14}$$

C. Implementation of sub-GOP and Parity Packet Allocation

Given the variables in (14) are integers and the mathematical complexity of the equations, we believe that it is impossible to get a closed-form solution to the optimization problem given by (14). Moreover, it is computational prohibitive to numerically get the global optimal solution by trying to evaluate $\bar{D}_{total}$ for all the possible allocation patterns, and selecting the one that leads to the minimal $\bar{D}_{total}$. In fact, when one GOP includes L P-frames, and the number of

[1] Note that $r_0 = 0$.

$$P(v_{k,r_{m+1}}) = \binom{R(r_{m+1})}{v_{k,r_{m+1}}} p_{k,r_{m+1}}{}^{R(r_{m+1})-v_{k,r_{m+1}}}(1-p_{k,r_{m+1}})^{v_{k,r_{m+1}}} \tag{9}$$

$$\mathcal{I}_{k,m+1}(z) = \{u_{k,j}, v_{k,r_{m+1}} \mid (\textstyle\sum_{j=r_m+1}^{r_{m+1}} u_{k,j}) + v_{k,r_{m+1}} = z, \forall u_{k,j} \in Z^*, \forall v_{k,r_{m+1}} \in Z^*\} \tag{10}$$

$$\bar{d}_{m+1}(k) = \textstyle\sum_{z=N-K+1}^{N} \sum_{\mathcal{I}_{k,m+1}(z)} \left(\left(P(v_{k,r_{m+1}}) \prod_{j=r_m+1}^{r_{m+1}} P(u_{k,j}) \right) \left(\sum_{j=r_m+1}^{min(k,r_{m+1})} u_{k,j} f(k-j)\bar{d} \right) \right) \tag{11}$$

$$\bar{d}'_{m+1}(k) = \begin{cases} \bar{d}_{m+1}(k) & \text{if } k \le r_{m+1} \\ \sum_{z=N-K+1}^{N} \sum_{\mathcal{I}_{r_{m+1},m+1}(z)} \left(\left(P(v_{r_{m+1},r_{m+1}}) \prod_{j=r_m+1}^{r_{m+1}} P(u_{r_{m+1},j}) \right) \left(\sum_{j=r_m+1}^{r_{m+1}} u_{r_{m+1},j} f(k-j)\bar{d} \right) \right) & \text{if } k > r_{m+1} \end{cases} \tag{16}$$

RS parity packets is R, there would be $\binom{L+R-1}{R}$ possible allocation patterns to be investigated. For example, if $L = 30$ and the number of RS parity packets is 40, there would be $\binom{69}{40} = 2.39 \times 10^{19}$ possible allocation patterns. Obviously, calculating the value of D_{total} using (13) for all the 2.39×10^{19} allocation patterns would be unfeasible.

Since it is computational prohibitive to determine the global optimal solution, we propose to simplify this process by firstly partitioning frames into sub-GOPs, and secondly allocate the redundancy among those sub-GOPs.

1) The first stage will be executed before the video encoding process, where the frames partition problem will be solved using a greedy algorithm that finds the local optimal size for each sub-GOP. In other words, the local optimal size for the first sub-GOP will be determined and then the second sub-GOP and so on. The sub-GOP size that minimizes the expected distortion *per frame* will be selected. To do this, the algorithm will try all the possible sizes starting from 1, and for each candidate the total expected distortion caused by this sub-GOP will be evaluated using (12), and then it will be divided by the size of the sub-GOP, so as to obtain the expected distortion *per frame*. Finally, the sub-GOP size that leads to the smallest expected distortion *per frame* will be chosen. After that, the algorithm will determine the local optimum sizes of the following sub-GOPs by repeating the previous process.
2) The second stage, that aims at allocating RS parity packets for each sub-GOP, is executed in parallel with the video encoding process. For this reason the exact number of slices per frame, $S(i)$, will be known exactly at the end of the video encoding process of each frame i. Thus, based on this knowledge, and the known positions where parity packets will be inserted (from the first stage), the actual number of parity packets will be decided on the fly, using the following equations:

$$R(r_k) = \begin{cases} \left\lceil \mu \sum_{i=1}^{r_1} S(i) \right\rceil & k = 1 \\ \left\lceil \mu \sum_{i=1}^{r_k} S(i) \right\rceil - \sum_{i=1}^{k-1} R(r_i) & k > 1 \end{cases} \tag{15}$$

with $R(r_k)$ being the amount of parity packets to be inserted at frame r_k. It is worth reminding that this position has been already determined in the first stage. The operation $\lceil X \rceil$ is used to get the minimum integer number greater than or equal to X.

D. Implementation of Reference Buffer Updating

At the video decoder side, the reference buffer updating technique can efficiently stop error propagations at the expense of high computational complexity. One solution to reduce the complexity is using the slice level reference buffer updating instead of frame level updating. This approach works by keeping track of all the slices that use a previously unavailable slice as reference, those slices will be called *dependant* slices in the following. This prediction information could be obtained by exploiting the motion vectors at the decoder side. In such a way to build a tree structure that describes the prediction dependency between slices. So once a slice is re-decoded and updated, all the slices whose tree root is this node will be also re-decoded and updated. Using this slice level updating technique can reduce the computational complexity without sacrificing the error resilient performance.

In order to further lower the computational complexity of the reference buffer updating in the RVS-LE approach, another solution is that the *late-arrival* packets can only be exploited within the current sub-GOP, whereas the *late-arrival* packets of the previous sub-GOP are simply discarded. For this low complexity solution, the sub-GOP size and the parity packet number for each sub-GOP should be allocated in a different way because the total expected distortion (13) needs to be calculated differently. In this case, the term $\bar{d}_{m+1}(k)$, all the distortion in frame k that is caused by the $(m+1)th$ sub-GOP, should be evaluated as (16) instead of (11). The main difference between (16) and (11) is that for any packets belonging to sub-GOP $m+1$, if they arrive later than the frame r_{m+1}, they will not be exploited for updating. The remaining process of determining the sub-GOP size and parity packet number is the same, so we name this scheme as Simplified RVS-LE.

E. A Benchmark: Frame-Level Evenly FEC

In order to evaluate the performance of the proposed RVS-LE approach, frame level FEC coding is provided as a benchmark. For the delay constraint FEC video coding, one possible approach is to perform RS coding at frame level, which means that the RS coding block contains data packets from the same frame. Let us assume that, for the i-th frame in one GOP there are $S(i)$ source packets and $R(i)$ RS parity packets, and that we want to evenly allocate the parity packets over all the GOP frames, taking into account that $R(i)$ should be an integer and the value $S(i)$ varies from frame to frame, so $R(i)$ can be evaluated as $R(i) = \lceil \mu S(i) \rceil$, where μ is the intended parity packet rate of RS coding. For this approach, the RS parity packets are evenly allocated among all the frames, from now on we name this approach as Evenly FEC. It is important to note that, when $S(i)$ and μ are small, at least one RS parity packet will be allocated. Let us take one example, $S(i) = 1$ and $\mu = 0.2$, then $R(i)$ is 1, in this case, the Evenly FEC approach degrades to duplicating the source packets.

It is worth noting that, with the fast algorithm of the proposed RVS-LE approach, the number of parity packets for each sub-GOP is proportional to the number of source packets within it. Therefore, unequal error protection scheme based on the frame position of the GOP is not applied here. This is why the Evenly FEC scheme is chosen as our benchmark.

TABLE I
AVERAGE DELAY AND THE RATE OF UNRECEIVED PACKETS ON TIME USING DIFFERENT MAXIMUM END-TO-END DELAY, FOR PACKET LOSS AND DELAY PATTERNS FILE 3,10 AND 20 IN Q15-I-16R

File	average delay	the rate of unreceived packets on time (%), with different maximum end-to-end delay				
		infinity	350 ms	300 ms	250 ms	200 ms
$f3$	125 ms	3.25	3.25	3.25	3.25	3.33
$f10$	160 ms	11.38	11.68	13.41	18.16	26.56
$f20$	160 ms	20.68	21.08	22.49	27.00	33.90

TABLE II
THE EFFECT OF $\overline{S}$ ON THE OPTIMIZATION PROCESS

video sequence	$\overline{S}$	sub-GOP size	$\overline{PSNR}(dB)$
Foreman (QP = 26)	6 (actual $\overline{S}$)	5 4 4 4 4 4 3 1	37.14
	3	5 5 5 5 5 4	36.70
	10	4 4 4 4 4 3 3 3	36.96
Coastguard (QP = 32)	7 (actual $\overline{S}$)	4 4 4 4 4 4 4 1	31.44
	3	5 5 5 5 5 4	31.34
	12	4 4 4 4 3 3 3 3 1	31.36

IV. SIMULATION RESULTS

Our simulation setting is built on the JM14.0 [32] H.264/AVC codec. CIF video sequences Foreman, Coastguard and Stefan are used for the simulations, with various levels of motion. The GOP structure is IPPP, the frame rate is 30 frames per second, and the GOP length is 30 frames. The reference frame number is 1, in other words, only the previous frame is used for prediction. As for packetization, one slice per packet is used, and taking the MTU of wireless network into account, we set the target slice length as 400 bytes. The average luminance PSNR is used to assess the objective video quality, and the mean PSNR ($\overline{PSNR}$) is averaged over 100 trials. Packet loss and delay patterns for the Internet experiments specified in Q15-I-16r [33] are used to emulate the real Internet environments. Table I lists the average delay and the rate of unreceived packets on time using different maximum end-to-end delay, for the file 3, 10 and 20 in Q15-I-16r. Those three files are used for simulation in this paper, and we will refer to them as f3, f10 and f20. It is worth noticing that the received packet rate for the *infinity* end-to-end delay is equivalent to the *physical* packet loss rate. Moreover, we assume the channel conditions are available at the encoding side. For the unrecoverable lost packets, the *motion copy* error concealment algorithm implemented in JM14.0 [32] decoder is used. In the following simulations, we set the maximum delay to 300 ms, unless otherwise noted. This value is within the acceptable maximum delay for real-time video communications, e.g., video conferences, which is 150-400 ms [3].

Given that the two parameters $\overline{S}$ and α are used in the optimization process, and because they may not be accurately determined, in the first set of simulations we will investigate the impact of these two parameters on the final results. In Table II, we report the obtained $\overline{PSNR}$, along with the obtained pattern of sub-GOP size, for Foreman and Coastguard sequence with different values for $\overline{S}$, with the file f10, and $\mu = 40\%$. For the Foreman sequence we used QP= 26, and we tested $\overline{S} = \{3, 6, 10\}$ with 6 being the actual value of $\overline{S}$; for the Coastguard sequence we set QP to 32, and we tested $\overline{S} = \{3, 7, 12\}$ with 7 being the actual value of $\overline{S}$. From this table we could see the impact of using inaccurate $\overline{S}$ on the final results, and in particular we could see that when the used $\overline{S}$ is smaller than the actual value then the sub-GOP size will enlarge; whereas when the used $\overline{S}$ is larger than the actual $\overline{S}$, the sub-GOP size will diminish. This happens, because small $\overline{S}$ requires the inclusion of more frames, in each sub-GOP, to improve the performance of the RS code; whereas, for large $\overline{S}$ small sub-GOP size becomes more suitable, and in this case, the beginning frames of the sub-GOP could be protected properly for small sub-GOP size. We can also notice that when the used $\overline{S}$ is the same as the actual $\overline{S}$, it leads to the best performance, and this demonstrates the effectiveness of the proposed sub-GOP and RS parity packet allocation algorithm, and the correctness of the optimization framework. One more conclusion that we can come to, from this results, is that despite the huge mismatch between the used $\overline{S}$ and its actual value, the average PSNR impairment is limited to 0.44, and 0.1dB for Foreman and Coastguard sequence, respectively. It is important to note that for practical application the expected mismatch between the estimated and actual value of $\overline{S}$ would be less than those reported in Table II. Therefore, based on these results, we could conclude that using the predicted $\overline{S}$ instead of the actual $\overline{S}$ will lead to nearly optimal performance. Secondly, for the distortion attenuation function $f(n) = \alpha^{n-1}$, Fig.3 shows the effects of different values of α. As shown in this figure, α will not affect the performance hugely. Based on this observation, α will be set to one, this is equivalent to say that the propagated distortion does not attenuate.

In the second set of simulations, we study the effects of different rates of the inserted parity packets, μ, on the performance. The file f10 is used in this simulation. We try different values of μ, including 25%, 30%, 40%, 50% and

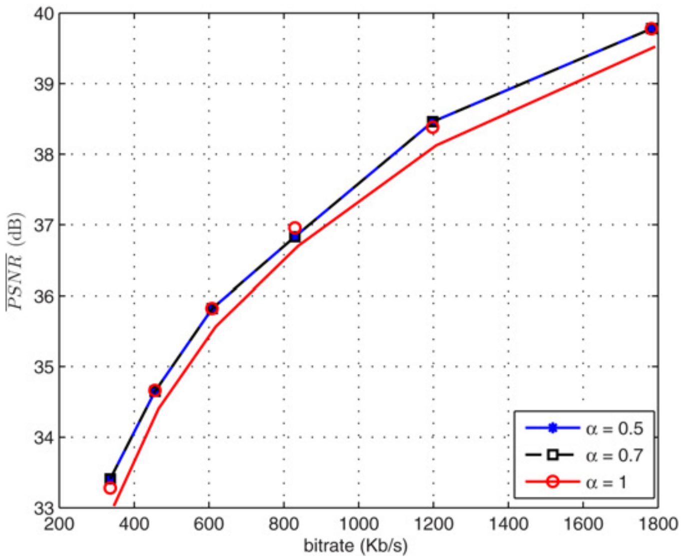

Fig. 3. The effects the parameter α, Foreman CIF sequence, maximum delay is 300 ms.

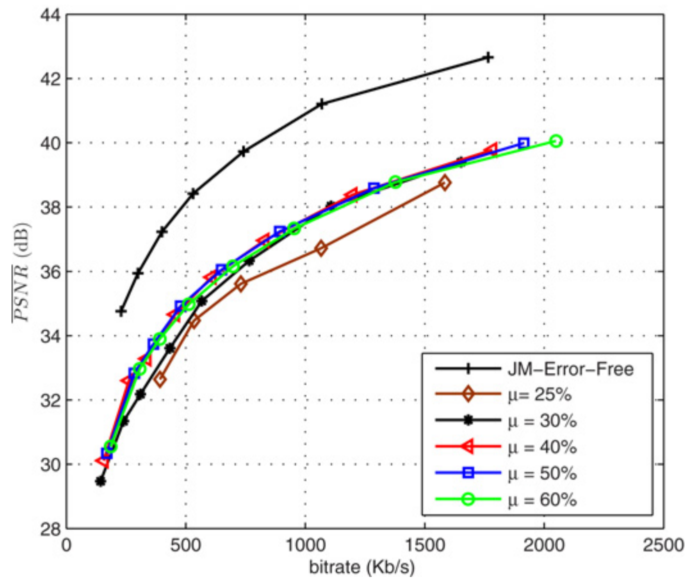

Fig. 4. Average PSNR versus bitrate for various parity packet rates μ (redundancy), CIF Foreman sequence is used, packet loss and delay pattern file is f10.

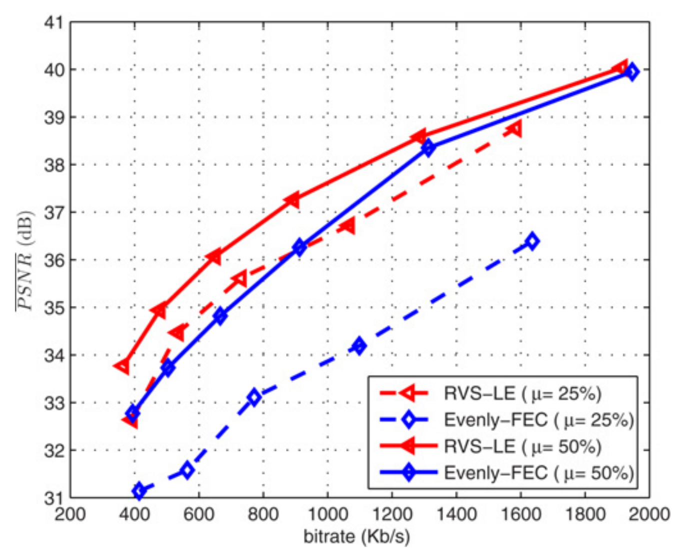

Fig. 5. Average PSNR versus bitrate for improper parity packet rates μ = 25% and μ = 50%, CIF Foreman sequence is used, packet loss and delay pattern file is f10.

60%. Simulations with various quantization parameters (QP) are carried out to span a considerable bitrate range. Fig.4 shows the average PSNR versus bitrate with different μ, for Foreman sequence. In general, the $\overline{PSNR}$ curve for μ = 25% and μ = 30% is much lower than other cases, whereas, the $\overline{PSNR}$ curves for μ = 60% is slightly lower than that of 40% and 50%, with the performance of 40% and 50% is almost equivalent. Consequently, in the following simulations, we use μ = 40% for the file f10, because lowering the parity packet rate can improve the performance for the error free case. By doing similar simulations for different average packet loss rate, it is found that the RS parity packet rate, μ, should be proportional to the average packet loss rate, p. In this article, $\mu = \psi p + \phi$ is used, with $\psi = 2$ and $\phi = 0.2$. This kind of linear FEC redundancy allocation method was also applied in the implementation of Skype [34]. So in the following simulations, 26% and 60% RS parity packet rate are used for the file f3 and f20, respectively. It is important to note that, even with improper μ, the proposed method still outperforms the Evenly-FEC method. Fig.5 shows the average PSNR versus bitrate curves for f10 with improper parity packet rates μ = 25% and μ = 50%. For the case with low redundancy μ = 25%, its gain over Evenly-FEC is larger than that with high redundancy μ = 50%. This phenomenon indicates that using the sub-GOP concept is more important when the inserted FEC redundancy is low.

In Fig. 6, the effect of the proposed sub-GOP allocation is studied by comparing its performance with the empirical setting of the sub-GOP size as 1 and 2. In all the approaches, *late-arrival* packet update has been applied. It is important to note that, when the sub-GOP size is 1, each frame has its own parity packets, and therefore in comparison with the sub-GOP approach the RS decoder does not have to wait for the parity packets allocated at several frames later. However, and in spite of that, its performance is not as good as that of the proposed RVS-LE. This is because when the sub-GOP size is small, the performance of the RS code is low. Also when the sub-GOP size is set to 2 the performance would be lower than RVS-LE, however, in this case the performance gap will be smaller, this is because the average sub-GOP length for the RVS-LE is 4.

In Fig. 7, we compare the proposed approach with the Hybrid MDC with sophisticated error concealment approach proposed in [15]. In order to have fair comparison, in this simulation, Bernoulli channel model with 10% of packet loss rate, and without delay, has been used. Moreover, the RTP/UPD/IP header length 40 byte is accounted in the bitrate of the RVS-LE approach. From the reported results we can notice that the proposed approach outperforms the Hybrid MDC, by nearly 2 to 5dB, and the RS-MDC by 3 to 6 dB.

In Fig. 8 and Fig. 9, the average PSNR versus bitrate curves are plotted for packet loss and delay pattern in file f20 and

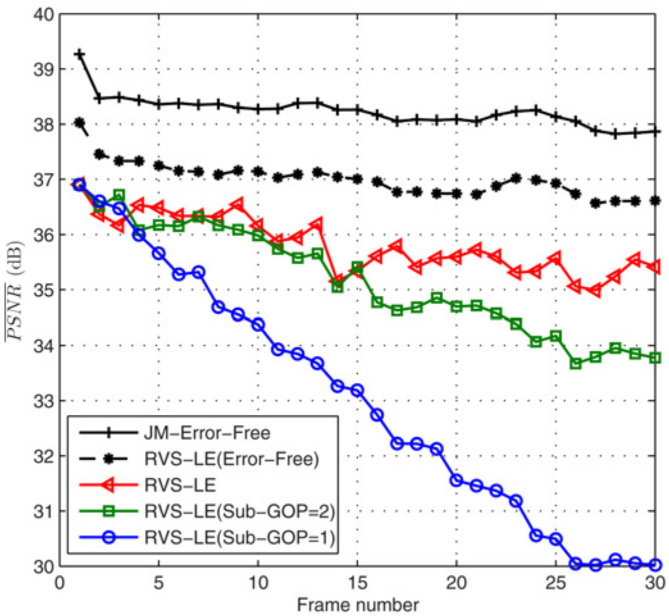

Fig. 6. Effects of sub-GOP size, Foreman CIF sequence, QP = 28 (608.64 kbps), packet loss and delay pattern file is f10.

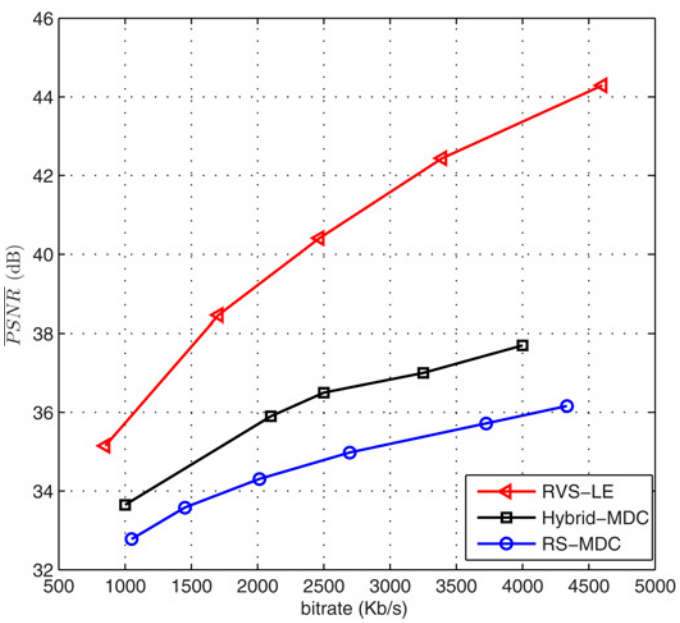

Fig. 7. PSNR versus bitrate for RVS-LE, Hybrid-MDC and RS-MDC; Bernoulli 10% packet loss rate without delay; CIF Foreman sequence, GOP length 30.

f10, respectively. We compare the proposed RVS-LE with other approaches, including the simplified RVS-LE described in Section III-D, Evenly-FEC, Evenly-FEC (No Update) and RS-MDC [11]. For the Evenly-FEC and Evenly-FEC (No Update), parity packets are allocated as described in Section III-E. However, the two approaches differ at the decoder side, with the former exploiting the *late-arrival* packets to update the reference buffer, and the latter simply discards them. As for the RS-MDC, it also discards the *late-arrival* packets. It is interesting to note that, in all cases the RVS-LE approach outperforms the RS-MDC approach significantly. Moreover, the comparison of both RVS-LE and Evenly-FEC, on one side, with the simplified RVS-LE and Evenly-FEC (No Update), on the other side, shows the importance of using all the *late-arrival* packets, even if they are belonging to previous sub-GOPs. In the whole bitrate range in Fig. 8 and low bitrate range in Fig. 9, the simplified RVS-LE has similar or higher performance than the Evenly-FEC, which means although the simplified RVS-LE only exploits the *late-arrival* packets of the current sub-GOP, it recovers most of the unavailable packets at low bitrate, because at this range of bitrate the simplified RVS-LE have relative large sub-GOPs. This, consequently, makes its performance comparable to those of Evenly-FEC, that exploits all the *late-arrival* packets. In Fig.9, and at high bitrate we notice that the performance of the simplified RVS-LE deteriorates in comparison with evenly-FEC, this is because the sub-GOPs tend to be small at high bitrate, and consequently more and more *late-arrival* packets will not be recovered. This effect does not happen in Fig.8, this is because the inserted parity packets are much higher, with f20 than those used for f10.

To evaluate the performance of the proposed approach, for relatively good channel condition, in Fig.10 we report the average PSNR versus bitrate for the packet loss and delay pattern in file f3. The reported results are for the following approaches: JM-Error-Free, RVS-LE, simplified RVS-LE, Evenly-FEC, and Evenly-FEC (No Update). It is worth noticing that for the packet loss and delay pattern in file f3, all the packets arrive at the destination before the 300 ms display deadline (this could be seen in Table I), therefore the performance of RVS-LE is similar to the simplified RVS-LE, and Evenly-FEC is also similar to Evenly-FEC (No Update). Thus, in this case where no packet arrive after its display deadline, these results serve to show the effectiveness of the sub-GOP based approach in recovering the physically lost packets.

In order to validate the proposed approach with respect to the maximum allowed end-to-end delay, in Fig.11 we show the average PSNR versus bitrate for different maximum end-to-end delays, namely 200, 250, 300, and 350 ms. These results have been obtained using the same parity packet rate, and the loss and delay pattern in file f10. From these results we could notice that the smaller the allowed delay is, the lower the expected PSNR would be. It is worth noticing that with 200 ms end-to-end delay, the unreceived on-time packet rate is 26.56%, this would have been a big loss rate for a traditional applications that discard the *late-arrival* packets, however, with the proposed approach we could still achieve 33.16dB at 1.2 Mbps bitrate.

Re-decoding and updating the reference buffer will increase the computational complexity at the video receiver side. In Table III, the ratio of slice number that need re-decoding and updating to the total slice number is reported for the Foreman and Coastguard video sequences, where the packet loss and delay pattern is file f10, the parity packet rate is 40%, and the maximum end-to-end delay is set to 300 ms and 200 ms. Two implementation methods with different complexity are used, including RVS-LE using frame-level and slice-level reference buffer re-decoding and updating. From the table it

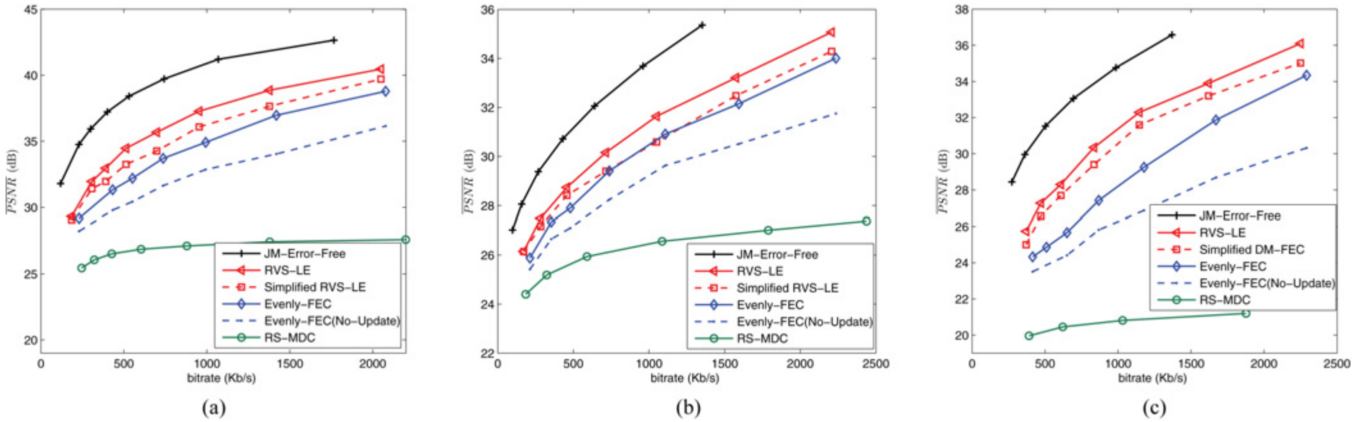

Fig. 8. $\overline{PSNR}$ versus bitrate for different approaches, the packet loss and delay pattern is file f20; (a) Foreman, (b) Coastguard, (c) Stefan.

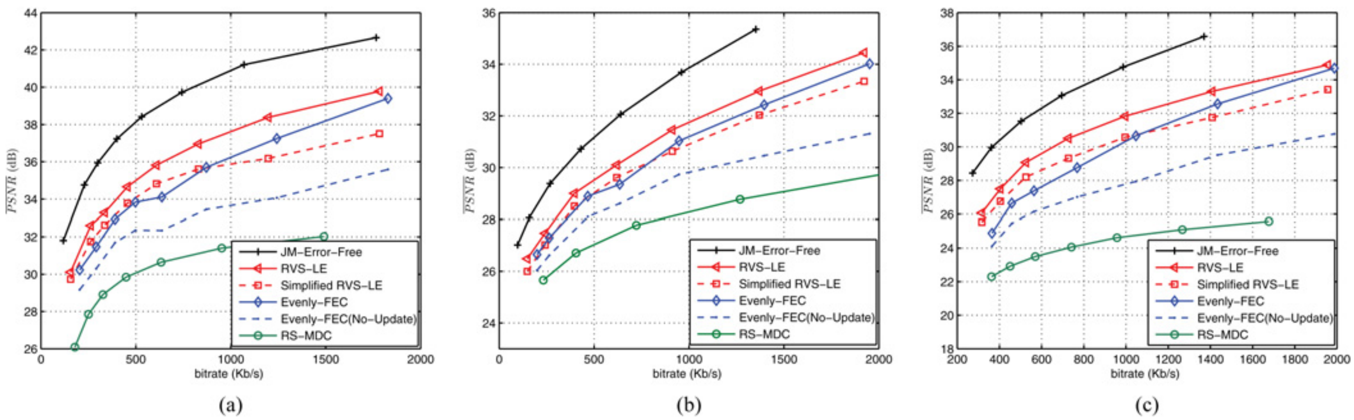

Fig. 9. $\overline{PSNR}$ versus bitrate for different approaches, the packet loss and delay pattern is file f10; (a) Foreman, (b) Coastguard, (c) Stefan.

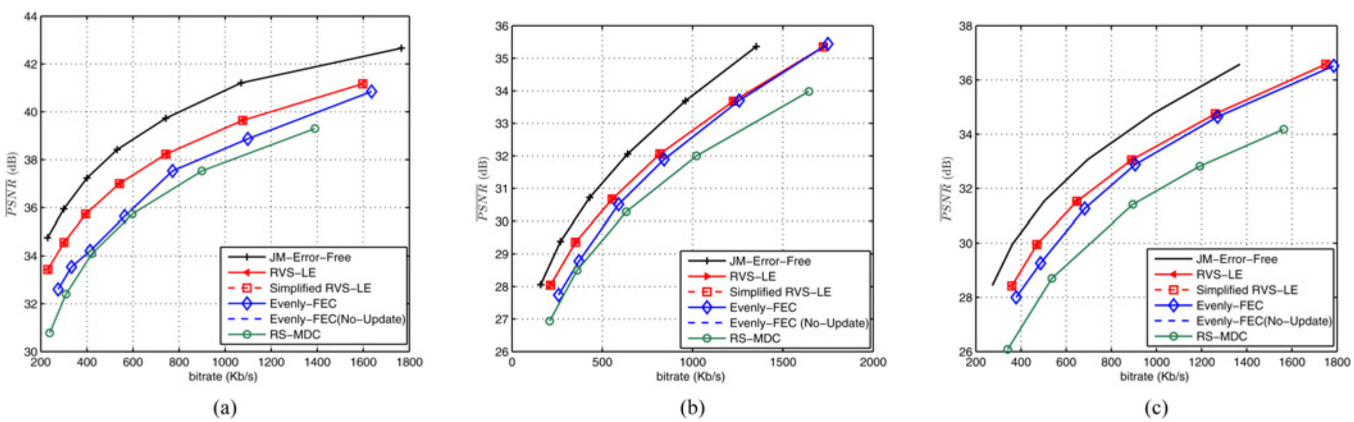

Fig. 10. $\overline{PSNR}$ versus bitrate for different approaches, the packet loss and delay pattern is file f3; (a) Foreman, (b) Coastguard, (c) Stefan.

is observed that, with 300 ms maximum end-to-end delay, the average ratio of slices that need re-decoding is 0.131 for the scheme with frame-level re-decoding and updating, whereas for the one with slice-level re-decoding is 0.063. When the maximum end-to-end delay is stringent, i.e., T_{max} = 200, this ratio is around 0.4 by using the slice-level updating technique. This information demonstrates that the reference buffer re-decoding and updating technique does not require tremendous computational resource, especially for advanced slice-level updating technique. Another observation is that, slice-level updating technique is especially useful at high bitrate, i.e., low QP value, this is because at high bitrate the slice number per frame is large, so frame-level updating will waste more computational resource.

V. Conclusions

In this paper, a real-time error resilient video streaming scheme exploiting the *late-arrival* packets and the out-of-order packets has been proposed. In the proposed approach, the *late-arrival* packets are not simply discarded, but they are used to boost the reconstructed video quality at the receiver side. In

TABLE III
THE RATIO OF SLICE NUMBER THAT NEED RE-DECODING AND UPDATING TO THE TOTAL SLICE NUMBER; PACKET LOSS AND DELAY PATTERN IS FILE F10, PARITY PACKET RATE IS 40%, MAXIMUM END-TO-END DELAY IS 300 MS AND 200 MS

Sequence	QP	T_{max} = 300 ms		T_{max} = 200 ms	
		frame-level	slice-level	frame-level	slice-level
Foreman	22	0.137	0.057	0.813	0.368
	26	0.125	0.059	0.735	0.376
	30	0.137	0.077	0.780	0.498
Coastguard	28	0.142	0.056	0.879	0.384
	32	0.121	0.057	0.758	0.383
	36	0.127	0.074	0.706	0.480
Average		0.131	0.063	0.778	0.414

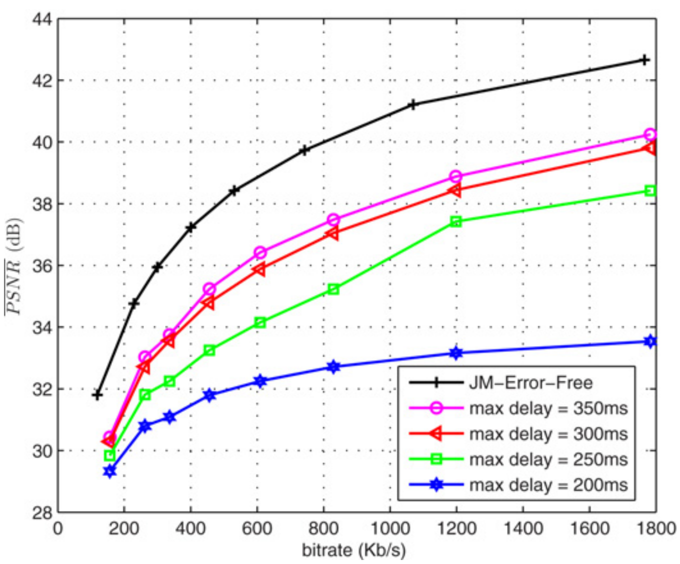

Fig. 11. Effects of the allowed maximum delay; CIF Foreman sequence, the packet loss and delay pattern is file f10, parity packet rate is 40%.

order to better exploit the *late-arrival* packets and the out-of-order packets, we propose to use packets of one sub-GOP, which contains a variable number of frames, as the RS coding block. Given the maximum end-to-end delay, the RS parity packet rate, the network packet loss rate and the packet delay distributions, a theoretical framework is presented to calculate the optimal sizes of the sub-GOPs and the amount of parity packets for each sub-GOP, so as to achieve the best error resilient performance. Since it is computational prohibitive to get the global optimal solution for the theoretical framework, a fast algorithm is also proposed for the practical applications. Meanwhile, a simplified version of the proposed scheme is also presented, where only the *late-arrival* packets within the current sub-GOP are exploited. In order to validate the proposed approach, its performance has been compared with other state-of-the-art real-time error resilient approaches in different environments. The experimental results demonstrate that the proposed approach outperforms the existing error resilient schemes significantly.

ACKNOWLEDGMENT

The authors would like to thank Associate Editor 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 Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 645–656, Jul. 2003.
[2] T. Stockhammer, M. Hannuksela, and T. Wiegand, “H.264/AVC in wireless environments,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 657–673, Jul. 2003.
[3] *Recommendation ITU-T G.114, One-Way Transmission Time*, ITU, 1996.
[4] W. Yao, S. Wenger, J. Wen, and A. Katsaggelos, “Error resilient video coding techniques,” *IEEE Signal Process. Mag.*, vol. 17, no. 4, pp. 61–82, Jul. 2000.
[5] A. Katsaggelos, Y. Eisenberg, F. Zhai, R. Berry, and T. Pappas, “Advances in efficient resource allocation for packet-based real-time video transmission,” *Proc. IEEE*, vol. 93, no. 1, pp. 135–147, Jan. 2005.
[6] R. Zhang, S. Regunathan, and K. Rose, “Video coding with optimal inter/intra-mode switching for packet loss resilience,” *IEEE J. Selected Areas Commun.*, vol. 18, no. 6, pp. 966–976, Jun. 2000.
[7] 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 Trans. Multimedia*, vol. 9, no. 3, pp. 445–454, 2007.
[8] S. Soltani, K. Misra, and H. Radha, “Delay constraint error control protocol for real-time video communication,” *IEEE Trans. Multimedia*, vol. 11, no. 4, pp. 742–751, Jun. 2009.
[9] P. Chou and Z. Miao, “Rate-distortion optimized streaming of packetized media,” *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 390–404, Apr. 2006.
[10] 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 *Proc. IEEE ICME*, Aug. 2001, pp. 96–99.
[11] T. Tillo, M. Grangetto, and G. Olmo, “Redundant slice optimal allocation for H.264 multiple description coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 1, pp. 59–70, Jan. 2008.
[12] C. Zhu, Y.-K. Wang, M. Hannuksela, and H. Li, “Error resilient video coding using redundant pictures,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 1, pp. 3–14, Jan. 2009.
[13] I. Radulovic, P. Frossard, Y.-K. Wang, M. Hannuksela, and A. Hallapuro, “Multiple description video coding with H.264/AVC redundant pictures,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 1, pp. 144–148, Jan. 2010.
[14] 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 J. Appl. Signal Process.*, vol. 2006, no. 1, p. 12, 2006.
[15] W.-J. Tsai and J.-Y. Chen, “Joint temporal and spatial error concealment for multiple description video coding,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 12, pp. 1822–1833, Dec. 2010.
[16] R. Liu, Y. Wang, and Y. Zhang, “A frame-related approach for performance improvement of MPE-FEC in DVB-H,” *IEEE Trans. Broadcast.*, vol. 57, no. 4, pp. 888–894, Dec. 2011.
[17] D. Gomez-Barquero, D. Gozalvez, and N. Cardona, “Application layer FEC for mobile TV delivery in IP datacast over DVB-H systems,” *IEEE Trans. Broadcast.*, vol. 55, no. 2, pp. 396–406, Jun. 2009.
[18] E. Baccaglini, T. Tillo, and G. Olmo, “Slice sorting for unequal loss protection of video streams,” *IEEE Signal Process. Lett.*, vol. 15, pp. 581–584, Dec. 2008.
[19] T. Tillo, E. Baccaglini, and G. Olmo, “Unequal protection of video data according to slice relevance,” *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1572–1582, Jun. 2011.
[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 Trans. Multimedia*, vol. 7, no. 4, pp. 753–765, Aug. 2005.
[21] F. Zhai, Y. Eisenberg, T. Pappas, R. Berry, and A. Katsaggelos, “Rate-distortion optimized hybrid error control for real-time packetized video transmission,” *IEEE Trans. Image Process.*, vol. 15, no. 1, pp. 40–53, Jan. 2006.
[22] G. Tan and T. Herfet, “Optimization of an RTP level hybrid error correction scheme for DVB services over wireless home networks under strict delay constraints,” *IEEE Trans. Broadcast.*, vol. 53, no. 1, pp. 297–307, Mar. 2007.

[23] M. Wu, S. Makharia, H. Liu, D. Li, and S. Mathur, "IPTV multicast over wireless LAN using merged hybrid ARQ with staggered adaptive FEC," *IEEE Trans. Broadcast.*, vol. 55, no. 2, pp. 363–374, Jun. 2009.
[24] I. Rhee and S. Joshi, "Error recovery for interactive video transmission over the internet," *IEEE J. Selected Areas Commun.*, vol. 18, no. 6, pp. 1033–1049, Jun. 2000.
[25] M. Ghanbari, "Postprocessing of late cells for packet video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 6, pp. 669–678, Dec. 1996.
[26] J. Xiao, T. Tillo, C. Lin, and Y. Zhao, "Real-time forward error correction for video transmission," in *Proc. IEEE VCIP*, Nov. 2011, pp. 1–4.
[27] J. Xiao, T. Tillo, C. Lin, and Y. Zhao, "Dynamic sub-GOP forward error correction code for real-time video applications," *IEEE Trans. Multimedia*, vol. 14, no. 4, pp. 1298–1308, Aug. 2012.
[28] Y. Xu and T. Zhang, "Variable shortened-and-punctured reed-solomon codes for packet loss protection," *IEEE Trans. Broadcast.*, vol. 48, no. 3, pp. 237–245, Sep. 2002.
[29] M. V. Nguyen, K. Ko, W. Lee, and H. S. Lee, "A new scheme to predict erasures for reed-solomon decoder in t-dmb receiver," *IEEE Trans. Broadcast.*, vol. 53, no. 2, pp. 530–538, Jun. 2007.
[30] N. Farber, K. Stuhlmuller, and B. Girod, "Analysis of error propagation in hybrid video coding with application to error resilience," in *Proc. ICIP*, vol. 2. 1999, pp. 550–554.
[31] B. Girod and N. Farber, "Feedback-based error control for mobile video transmission," *Proc. IEEE*, vol. 87, no. 10, pp. 1707–1723, Oct. 1999.
[32] [Online]. Available: http://iphome.hhi.de/suehring/tml/download/
[33] Y.-K. Wang, S. Wenger, and M. M. Hannuksela, *Common Conditions for SVC Error Resilience Testing*, JVT document P206, Aug. 2005.
[34] X. Zhang, Y. Xu, H. Hu, Y. Liu, Z. Guo, and Y. Wang, "Modeling and analysis of skype video calls: Rate control and video quality," pp. 1–1.

Jimin Xiao was born in Suzhou, China. He received the B.S. and M.Eng. degrees in telecommunication engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2004 and 2007, respectively. Since 2009, he has been pursuing the Ph.D. degree at the University of Liverpool, Liverpool, U.K. From 2007 to 2009, he was a Software Engineer with Motorola (China) Electronics, Ltd., Tianjin, China, and a System Engineer at Realsil (Realtek) Semiconductor Corporation. Since March 2013, he has been a Visiting Scholar in the CeMNet Laboratory, Nanyang Technological University, Singapore. His current research interests include video streaming, image and video compression, and 3-D video coding.

Tammam Tillo (M'05–SM'12) was born in Damascus, Syria. He received the Engineer Diploma degree in electronic engineering from Damascus University, Damascus, in 1994, and the Ph.D. degree in electronics and communication engineering from the Politecnico di Torino, Turin, Italy, in 2005. From 1999 to 2002, he was with Souccar for Electronic Industries, Damascus, Syria. In 2004 he was a Visiting Researcher at the EPFL, Lausanne, Switzerland, and from 2005 to 2008, he was a Post-Doctoral Researcher at the Image Processing Laboratory, Politecnico di Torino, and for a few months he was an Invited Research Professor at the Digital Media Laboratory, SungKyunKwan University, Suwon, Korea. In 2008, he joined Xi'an Jiaotong-Liverpool University (XJTLU), Suzhou, China, and he was promoted to full Professor in 2012. He is currently the Head of the Electrical and Electronic Engineering Department since September 2010 and he was the Acting Head of the Department of Computer Science and Software Engineering from April 2012 to March 2013, XJTLU university. His current research interests include robust image and video transmission, image and video compression, and hyperspectral image compression.

Chunyu Lin was born in Liaoning Province, China. He received the B.S. degree from Bohai University, Jinzhou, China, and the M.S. degree from Yanshan University, Hebei, China, in 2002 and 2005, respectively, and the Doctor degree from Beijing Jiaotong University, Beijing, China, in 2011. From 2009 to 2010, he was a Visiting Researcher at the ICT Group, Delft University of Technology, Delft, The Netherlands. From 2011 to 2012, he was a Post-Doctoral Researcher with the Multimedia Laboratory, Gent University, Gent, Belgium. His current research interests include image/video compression and robust transmission, 2D-to-3D conversion, and 3-D video coding.

Yungang Zhang received the Bachelor's and Master's degrees in computer science from Yunnan Normal University, Yunnan, China, in 2002 and 2005, respectively. He is currently pursuing the Doctoral degree in computer science at the University of Liverpool, Liverpool, U.K. He is currently with the Department of Computer Science and Software Engineering, Yunnan Normal University. His current research interests include pattern recognition, machine learning, and image analysis.

Yao Zhao (M'06–SM'12) received the B.S. degree from Fuzhou University, Fujian, China, in 1989 and the M.E. degree from Southeast University, Nanjing, China, in 1992, both from the Radio Engineering Department, and the Ph.D. degree from the Institute of Information Science, Beijing Jiaotong University (BJTU), Beijing, China, in 1996. He became an Associate Professor at BJTU in 1998 and a Professor in 2001. From 2001 to 2002, he was a Senior Research Fellow with the Information and Communication Theory Group, Faculty of Information Technology and Systems, Delft University of Technology, Delft, The Netherlands. He is currently the Director of the Institute of Information Science, Beijing Jiaotong University. He is currently leading several national research projects from 973 Program, 863 Program, the National Science Foundation of China. His current research interests include image ideo coding, fractals, digital watermarking, and content-based image retrieval. He was a recipient of the National Outstanding Young Investigator Award of China in 2010.