Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2010, Article ID 729695, 11 pages
doi:10.1155/2010/729695

Research Article

Embedding Protection Inside H.264/AVC and SVC Streams

Catherine Lamy-Bergot and Benjamin Gadat

THALES Land and Joint Systems, RCP/DT and EDS/SPM Departments, 92704 Colombes, France

Correspondence should be addressed to Catherine Lamy-Bergot, catherine.lamy-bergot@fr.thalesgroup.com

Received 30 March 2010; Revised 2 July 2010; Accepted 26 August 2010

Academic Editor: Liang Zhou

A backward compatible error-protection mechanism embedded into the H.264 (AVC or SVC) syntax is described. It consists of the addition into the H.264 bitstream of supplementary network abstraction layer (NAL) units that contain forward error-correction (FEC) data generated by a block error-correction code. The proposed mechanism allows to leave the original information bits and NAL units intact and does not rely on any side information or extra signalling coming from lower layers, ensuring backward compatibility with the standard syntax. Simulation results obtained with Reed-Solomon and Low-density parity check error-correcting codes show significant improvements for both erroneous and lossy transmission channel configurations.

1. Introduction

The H.264 standard, both in its nonscalable (AVC) [1] and in its more recent scalable version (SVC) [2] has been established to offer enhanced coding efficiency when compared to previous video-coding standards such as the still widely used MPEG-2 one. The aim of the standardization effort has been to establish a solution enabling transmission of more video (or of video of better quality) over as diverse conditions as Internet/LAN, TV broadcasting, or mobile wireless networks.

To cope with the loss/error conditions that may occur on those various networks type, the H.264 video-standard includes error resilience tools, such as picture segmentation, intra-placement on various levels, reference picture selection, data partitioning, flexible macroblock ordering, and so forth, [3]. These tools may however remain insufficient to offer a complete recovery of a corrupted stream, leading to degraded video rendering when in presence of very erroneous transmission conditions, such as the one occurring over wireless channels. Indeed, wireless channels rely on physical (PHY) layer protection by means of forward error-correction (FEC) and possible retransmissions (ARQ for Automatic Repeat reQuest) to provide reliable transmission over their unreliable communication medium, but bandwidth limitations or eventual real-time constraints can prevent the transmission to be fully reliable.

There have been different research efforts to overcome this issue, in particular via the introduction of error resilient coding mechanisms in the H.264 standard or via the usage of additional data transmission, by means of ARQ and/or FEC above the physical layer. The first type of mechanism consists in efficiently concealing the losses or errors due to the transmission over the channel, by embedding of data useful for error concealment into the video stream [4], or by intelligently encoding or decoding the stream [5, 6]. Those approaches however can only minimize the impact of the wireless channel, and not actually correct the errors or losses.

For a perfect or almost perfect rendering of the information transmitted, one needs to rely on the second type of mechanism, namely the FEC and/or ARQ approaches. Beside the original approach where application and radio were considered directly connected, allowing various declinations of error-correction codes applications after compression and before transmission over the channel (e.g., in [7]), one finds in the literature and recent standards solutions integrating the protocol layers. Some rely on the introduction of retransmissions (ARQ or hybrid-ARQ) at the data link level [8, 9], while other solutions propose to introduce error correcting capability in transport layer (as promoted by IETF FecFrame [10] group or with RTP-FEC approaches such as [11], or with the fountain codes AP-FEC approach promoted by 3GPP [12, 13]). A more general approach, belonging to the "joint source and channel coding" field, which aims at finely adapting the transmitted content as well as the protection applied to the considered source and transmission conditions, has also been considered, with eventually a combination of FEC applied at higher and PHY levels [14]. Those different solutions show interesting performance for multimedia delivery over erroneous or lossy channels, but present the drawback of requiring modifications below the application level, which may be difficult to implement in real life if only for fear of backward compatibility issues for already deployed networks.

In this paper, we propose to introduce error-correction capability inside the video stream itself, transparently to the lower layers by embedding it in supplementary network abstraction layer units. This FEC capability introduced at the emission side will allow an aware receiver to correct the eventual losses and errors remaining after the transmission. As such, the approach is valid first in the case of packet losses due to packets drops in not reliable transport protocols such as UDP, or due to timeout for more reliable ones such as TCP, and second in the case of both packet losses and errors due to partially CRC-protected transport protocols such as UDP-Lite and DCCP. Interestingly, this protection is not transport or transmission channel dependent: it can be applied for wired or wireless transmissions, and is compatible with any transport protocol. Furthermore, this protection can be applied either directly together with the compression operation (for immediate or delayed transmission of the stream) or generated later, as a separate operation, typically within a transcoder. The interest of implementing the redundancy insertion in a transcoder module is that the operation can then be performed both over precompressed streams, or on the fly in a proxy or in a relay node if the transmission conditions necessitate it. Similarly, the decoding process can be either embedded directly into the video decoder or performed by a transdecoder module that performs the reverse operation to the transcoder one. For sake of simplicity, in the paper we will describe the case where the operation is collocated to the compression and decompression operations.

This paper is organised as follows: Section 2 presents first the H.264 standard network abstraction layer organisation and its syntax, and describes the proposed new redundancy NAL units syntax and their functionality. Section 3 details the corresponding system processing for insertion of redundancy to protect H.264 streams. Section 4 then presents the simulation conditions used and the corresponding simulation results obtained. Finally some conclusions are drawn in Section 5 and perspectives are presented.

2. H.264 Standard NAL Syntax

2.1. NAL Structure. H.264 has been designed to be as network independent as possible. This is made possible by the introduction of encapsulation by means of a network adaptation layer (NAL) which contains the video-coding layer (VCL), as illustrated by Figure 1. The VCL consists of the result of the compression engine, which is the compressed video data itself. The NAL adapts this video data to various network conditions with a transport oriented approach.

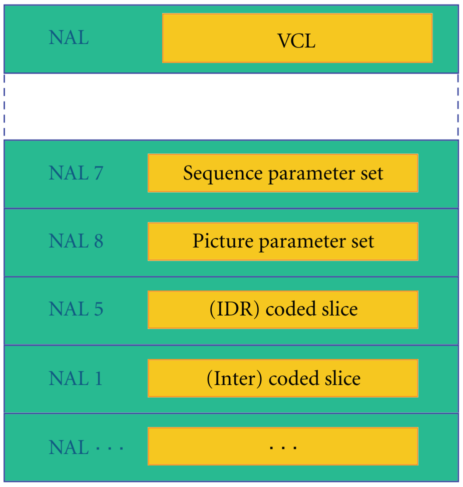


Figure 1: H.264 layer organization.

2.2. Inserting Redundancy by Means of Supplementary "Redundancy" NAL Units. The objective being to obtain a stream compliant with the H.264 specifications [1, 2] after the insertion of the redundant information, in order to have any standard decoder still decode the stream, it is necessary to first keep the original data information untouched, and secondly to place the protection in such a way a standard decoder will not try to interpret it. We propose to reach this goal by

(i) using systematic codes for FEC protection

(ii) inserting the redundancy into specific standard compatible NAL units.

As a matter of fact, introducing redundancy data in the stream by embedding it in specific NAL units will allow to respect the H.264 video-standard structure; which means that any standard compliant decoder will merely discard the supplementary information added for protection. FEC aware decoders will on the contrary extract the redundancy information in order to obtain corrected useful data.

Naturally, depending on the actual H.264 choice made, that is, either H.264/AVC or H.264/SVC, the NAL unit default header differs, which means that the implementation of the following mechanism must be attuned to the standard. In the following, we will detail the approach for H.264/AVC, with which it has been originally fully tested and validated. Nevertheless, except for the change of header (which is extended from the one-byte value in H.264 AVC to a four-bytes header including identifications but also error indication and importance information), the process of supplementary NAL units carrying redundancy information is identical.

In order to protect the carried information data, we propose to consider the case were several ($N \geqq 1$)

information data NAL units are used to generate several ($M \geqq 1$) redundancy NAL units. While the simple case of each unique information data NAL unit being followed by a unique redundancy data NAL unit is also being considered in our numerical results (see Section 4), only the more generic N/M case will allow to deal with bursts of losses or errors, but also will ensure that the overhead introduced by the supplementary NAL units is not too costly when high-rate protection is used.

As illustrated by Figure 2, the chosen data organisation is based on a matrix, composed of a first part which is filled line by line with the information data of the N considered video NAL units, and a second part which is filled by the redundancy information generated by reading the information data in columns. This setting allows for a line/column interleaving of the information data, the redundancy corresponding to a given information NAL unit being spread over up to M different redundancy NAL units.

In the case where block codes such as Reed-Solomon codes [15] or Low-Density Parity Check (LDPC) codes [16] are considered, the matrix row and column dimensions are, respectively, N' (with K' lines corresponding to the N information NAL units completed by eventual padding), and J, whose value is determined by dividing the overall size of the N video NAL units by the chosen code K' value. Then, J FEC encoding operations are done, resulting in $J*(N' - K')$ generated redundancy symbols that constitute the redundancy data to be transmitted.

This redundancy constitutes the pseudo VCL information of the supplementary redundancy NAL units, whose headers must then contain information allowing the decoder to regenerate the matrix even when some of the NAL units, whether information or redundancy ones, are lost.

2.3. Syntax for Redundancy NAL Units.

The proposed syntax for redundancy NAL units carries information necessary to allow the decoder to recover the original N data NAL units. In particular, the position indication information for each data NAL unit is provided, to ensure that the redundancy NAL units can be exploited even if one NAL unit is lost (meaning that in practice its size, varying by nature, is also lost).

The proposed syntax, for a new NAL unit type that we have in our system fixed to the value "30" then contains

(i) information on the type of redundancy NAL unit format (several can be considered, as illustrated later),

(ii) error protection code used (possibly an index from a predefined table),

(iii) additional (could be optional) information to allow for differentiation of frames corresponding to different matrixes (e.g., first video frame number),

(iv) numbers of data and redundancy NAL units: N, M,

(v) position of the N data NAL units carried by the matrix (this being stored in a (line, column) address format corresponding to the beginning of each NAL unit),

(vi) position of the M redundancy NAL units in the matrix (this being stored in a (line, column) address format corresponding to the beginning of each redundancy NAL unit),

(vii) a checksum (CRC) covering the whole NAL unit header.

Based on this list, it has been observed in practice, that depending on the size of the considered matrix, it was interesting to either place in each of the M redundancy NAL unit the whole description information, or separate the description information to reduce the cost in terms of used bits. The first solution, illustrated by the format (denoted type "01") and proposed in Figure 3, allows to easily deal with potential NAL unit losses, as all position and synchronisation information are repeated in each redundancy NAL unit, but leads to a prohibitive cost in terms of bit-rate if M is too large. The second solution, proposed to reduce the number of signalling bits, places the signalisation relative to the information data in a first NAL unit type (denoted type "00"), as illustrated by Figure 4, and then in each following redundancy NAL unit carrying the redundancy information place only the redundancy signalisation and possibly a reminder on the error protection code used, as illustrated by Figure 5, denoted type "10".

For comparison purpose, and to evaluate the interest of the interleaving matrix with respect to the header cost it introduces, we also considered a simplified 1/1 case where each data NAL unit is directly followed by a redundancy NAL unit. As a consequence, the headers have in this case been reduced to the minimal information of start code, standard NAL unit header and a very reduced extension including an index for the used error-correction code, the number of the video data frame protected (for simple loss detection) and a checksum, as illustrated by Figure 6.

3. System Processing for Insertion of Redundancy and Protection of H.264 Streams

As stated before, we will consider here the case where the protection mechanism is applied together with the compression process. Let us explain the proposed mechanisms both at the encoding and decoding side.

3.1. Encoding.

Beside its traditional tasks, the video encoder takes the information data (i.e., the video data NAL units) and feeds them into the systematic error-correction encoder to generate redundancy data, and then generates accordingly redundancy NAL units headers accordingly to the format given in Section 2.3, to produce the M redundancy NAL units. The process is detailed in Figure 7.

3.2. Decoding.

Beside performing its traditional tasks, the video decoder aware of possible redundancy NAL units presents also looks for such supplementary information. As detailed in Figure 8, the decoder first reeds NAL units in the bitstream and store them into its NAL units buffer up until finding a redundancy NAL unit, or reaching the buffer

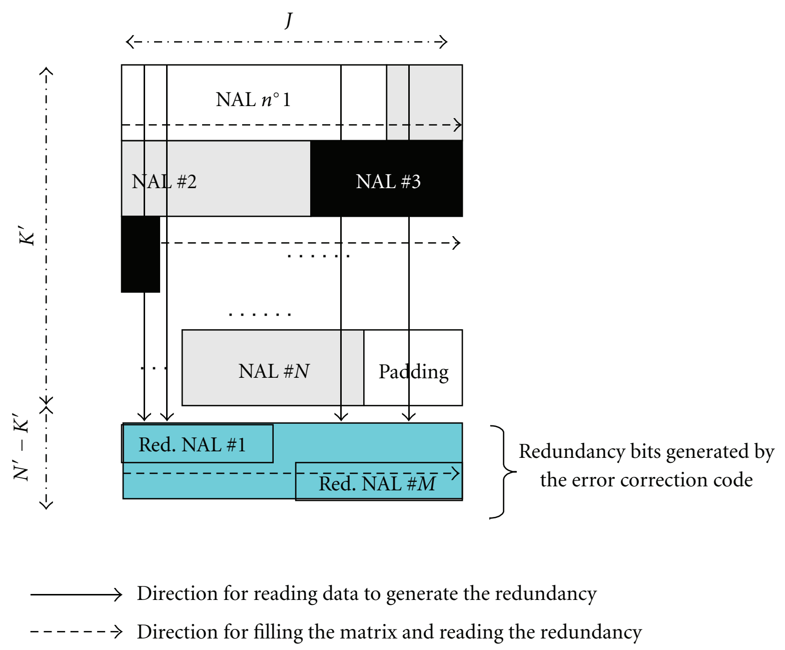

J
K′
N′ − K′
NAL n° 1
NAL #2
NAL #3
NAL #N
Padding
Red. NAL #1
Red. NAL #M
Redundancy bits generated by the error correction code
Direction for reading data to generate the redundancy
Direction for filling the matrix and reading the redundancy

Figure 2: N/M redundancy matrix organisation.

Code and redundancy matrix parameters — Signalisation for the considered information data — Signalisation for the generated redundancy

Start code (3 to 4 bytes)	Standard NAL header (1 byte)	NAL Red type "01"	Used code (index)	K'	$N' - K'$	J	Video data number (1st one)	Number of N video data NALs	@(line, column) $*N$	Redundancy counter (M)	@(line, column) $*M$	CRC	Redondance data

CRC coverage area

Figure 3: N/M NAL unit 30 proposed format, type "01".

Code and redundancy matrix parameters — Signalisation for the considered information data

Start code (3 to 4 bytes)	Standard NAL header (1 byte)	NAL Red type "00"	Used code (index)	K'	$N' - K'$	J	Video data number (1st one)	Number of N video data NALs	@(line, column) $*N$	CRC

CRC coverage area

Figure 4: N/M NAL unit 30 proposed format, type "00".

Code and redundancy matrix parameters — Signalisation for the generated redundancy

Start code (3 to 4 bytes)	Standard NAL header (1 byte)	NAL Red type "10"	Used code (index)	K'	$N' - K'$	J	Redundancy counter (M)	@(line, column)	CRC	Redondance data

CRC coverage area

Figure 5: N/M NAL unit 30 proposed format, type "10".

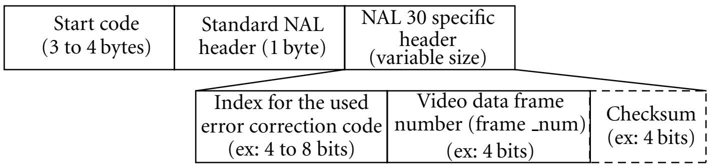

Start code (3 to 4 bytes)
Standard NAL header (1 byte)
NAL 30 specific header (variable size)
Index for the used error correction code (ex: 4 to 8 bits)
Video data frame number (frame _num) (ex: 4 bits)
Checksum (ex: 4 bits)

Figure 6: Simplified syntax of the supplementary NAL unit in case of 1/1 redundancy insertion.

Table 1: Backward compatibility tests for H.264 redundancy enhanced streams.

Considered sequence (including considered Reed-Solomon code)	PSNR after decoding with JM 16.2	PSNR after decoding with ffmpeg 0.5	PSNR after decoding with error correcting JM 12.1
Foreman QCIF, 128 kbps with an RS code (128, 64)	32.30	32.30	32.30
Mobile QCIF, 255 kbps with an RS code (255, 232)	30.93	30.93	30.93
Hall Monitor QCIF, 128 kbps with an RS code (255, 250)	39.91	39.91	39.91
Akiyo QCIF, 128 kbps with an RS code (255, 250)	43.34	43.34	43.34
Foreman CIF, 512 kbps with an RS code (255, 232)	35.17	35.17	35.17
Mobile CIF, 2048 kbps with an RS code (128, 64)	29.89	29.89	29.89
HallMonitor CIF, 512 kbps with an RS code (255, 232)	38.23	38.23	38.23
Akiyo CIF, 255 kbps with an RS code (255, 250)	41.06	41.06	41.06

maximal size (to cope with possible losses of NAL units or temporary absence of redundancy NAL units). With the first redundancy NAL unit received, the signalling information present in the NAL unit header allows to create the matrix at the decoding side, and then to launch the process of fetching all redundancy NAL units and corresponding bits. The redundancy decoding operation is performed when last redundancy NAL unit is received (or a data information NAL unit, which then leads to detecting loss of last(s) redundancy NAL unit and leads to loss of NAL unit processing step), allowing to generate the corrected information data, which is then used to update the data information NAL units, that are then send to the standard video decoder.

4. Numerical Results

4.1. Considered FEC Codes and Transmission Channel. The generation of the redundancy carried by the supplementary NAL units of type is obtained through error-correction codes, whose role is to provide forward error-correction capabilities in error-prone environments. Different families of codes exist, that can be more attuned to error or loss corrections, adapted to random impairments or bursty channels, or also more efficient for shorter or longer sizes of code blocks. The choice of the used error-correction code will consequently have to be made with respect to the characteristics of both the data to be transmitted and the transmission channel. The approach proposed in this paper was consequently made generic, to allow application to various error-correction codes, with the limitation of them needing to be systematic, to ensure that the information NAL units are transmitted unmodified. In our tests and simulations, we chose two different error-correction codes to illustrate the versatility of our solution. The first and primary example considered is one of the most well-known family of error-correction codes: Reed-Solomon (RS) codes [15], over Galois Field GF(2^8) to easily manipulate bytes. The codes will be referred to as RS(N', K') in the following, where N' is the codewords symbol length and K' the number of information symbols. The second example considered is the family of LDPC codes [16], built according to the progressive edge growth (PEG) algorithm [17]. Both RS and LDPC codes are applied in the following to any K' bytes of the upper part of the redundancy matrix to generate the $N'-K'$ redundancy bytes, that will then be used to generate the data part of the supplementary NAL units, as illustrated in Figure 2.

It is interesting to point out that the FEC approach can be used in conjunction with exiting robustness enhancement techniques already foreseen or used for H.264. Typically, even redundant slices, that are different from our supplementary slices in the sense where they operate duplication operations, and not error-correction, can be protected with the NAL unit type “30”. Similarly, the FEC can be used in conjunction with concealment techniques [18, 19] to deal with possible remaining errors by concealing them.

4.2. Backward Compatibility Compliance Tests. One of the interest of the presented approach is that the redundancy information is inserted in standard NAL units, using reserved numbers. As a consequence, an H.264 decoder that is not aware of the possibility to use the redundancy information contained in the supplementary NAL units, or that does not wish to use it (for instance due to complexity limitations considerations) will be in principle able to discard the supplementary packets and decode the stream.

To validate in practice the backward compatibility of our system, we have decided to test this capacity to skip supplementary NAL units, and for that have selected two

Table 2: Probability of decoding success for "Foreman" SVC encoded sequence (three layers) over a BSC channel.

SNR on the BSC	Observed BER	Decoding probability without Reed-Solomon encoding	Decoding probability with Reed-Solomon RS (128,120) encoding	Decoding probability with Reed-Solomon RS (128,120) encoding and header protection
5	$5 \cdot 10^{-3}$	**0**%	**0**%	**0**%
6.5	10^{-3}	**0**%	**0**%	**0**%
7.5	$5 \cdot 10^{-4}$	**0**%	**0**%	**14**%
8	10^{-4}	**0**%	**1**%	**71**%
9.5	10^{-5}	**0**%	**93**%	**100**%
10	$5 \cdot 10^{-6}$	5%	99%	100%
11	10^{-7}	71%	100%	100%

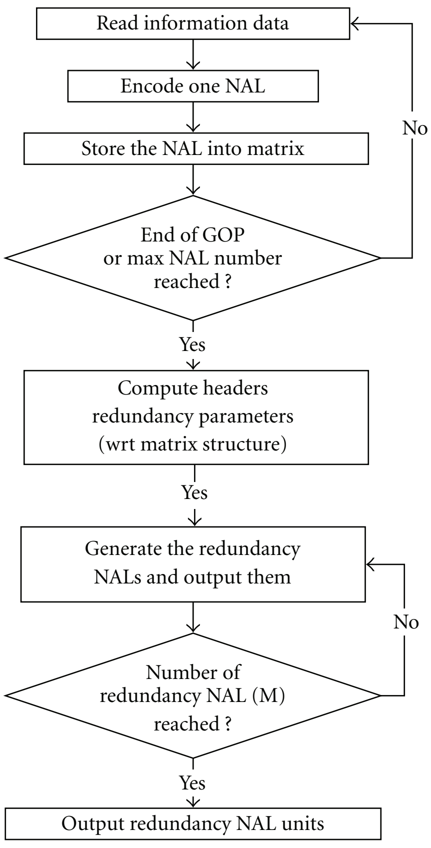

Read information data
Encode one NAL
Store the NAL into matrix
No
End of GOP or max NAL number reached ?
Yes
Compute headers redundancy parameters (wrt matrix structure)
Yes
Generate the redundancy NALs and output them
No
Number of redundancy NAL (M) reached ?
Yes
Output redundancy NAL units

Figure 7: *N*/*M* redundancy encoding process.

different decoders beside our own modified decoder. As our decoder, is based on the H.264 verification model version 12.1, the first reference decoder we chose is the current latest version of the verification model (JM 16.2). The second reference we selected is another well-known and well-used decoder of the online community: ffmeg (in its latest current version, ffmpeg-0.5), which include resilience tools, which may try to interpret the supplementary NAL units.

Simulations have been done with four different ITU-T reference sequences: "Foreman", "Akiyo", "Mobile Calendar" and "Hall Monitor", and using different Reed-Solomon codes, as detailed in Table 1. The results of this table, which presents the obtained PSNR for sequences including redundancy NAL units show that no difference can be observed between the three decoders in absence of perturbation. The impact of inserting redundancy by means of specific NAL units is consequently null in terms of backward compatibility.

4.3. H.264/AVC Tests. Again, simulations have been done with the four different ITU-T reference sequences: "Foreman", "Akiyo", "Mobile Calendar" and "Hall Monitor". In order to propose fair comparaisons, the different simulations were systematically done for the same overall throughput for compressed (in monoslice mode) and protected video. On the other hand, in order to offer diversity, various compression rates and temporal and spatial resolutions were considered. Similarly, two types of channel models have been considered, corresponding to either packet losses or errors. The first is a packet erasure channel (PEC), working at the NAL unit level, that emulates losses over wired channel as Internet or caused by an imperfect wireless channel. The second is a binary symmetric channel (BSC) corresponding to the case where the different protocol layers interconnecting application and radio are accepting bit errors thanks to partially CRC-protected transport and data link protocols. The results presented in Figure 10 to Figure 15 have been obtained with QCIF (176×144 pixels) spatial resolution considered at 15 Hz with I_1P_{14} format and CIF (352×288 pixels) spatial resolution considered at 30 Hz with I_1P_{29} format. Total throughput (including redundancy) for QCIF sequences was set to 128 kb/s except for "Mobile Calendar" which used 255 kb/s. Total throughput for CIF sequences was 512 kb/s except for "Mobile Calendar" which used 2048 kb/s and "Akiyo" which used 255 kb/s. Finally, the comparison presented in Figure 16 has been made with QCIF spatial resolution at 30 Hz, with I_1P_{14} format, for an overall throughput of 128 kb/s for "Foreman" sequence and 255 kb/s for "Mobile Calendar" sequence. Beside Reed-Solomon codes, we also have been using, in the case of the BSC channel, an irregular LDPC code of rate 1/2, operating over 512 input bits, that is, with $K' = 64$ and $N' = 128$. This irregular LDPC code has been designed with the PEG algorithm [17] using a maximal degree of 15, our purpose being to favour the frame Error-Rate which is of interest

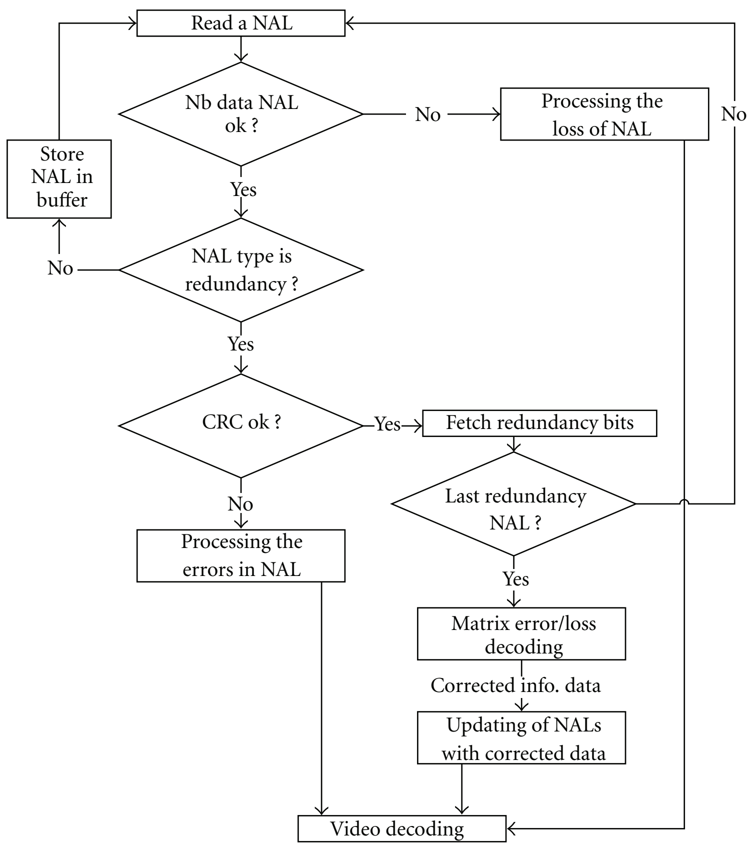


FIGURE 8: N/M redundancy decoding process.

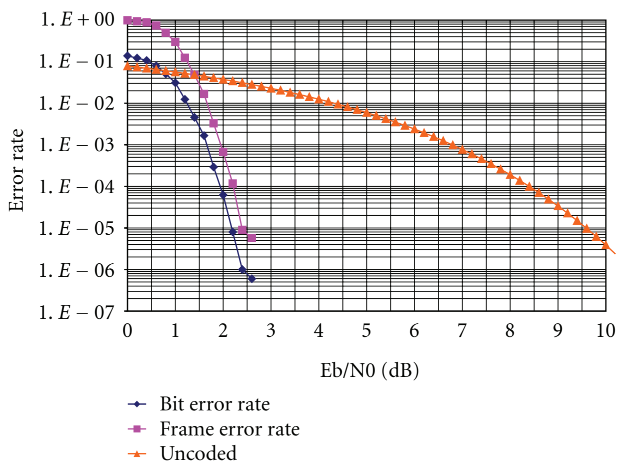


FIGURE 9: Error-Rate performance curves for the considered (128,64) irregular LDPC code.

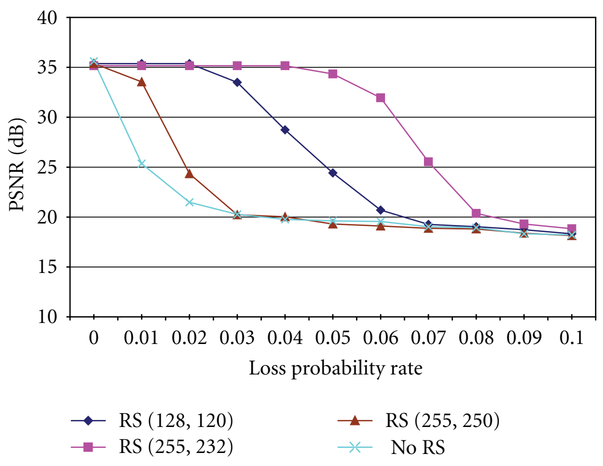
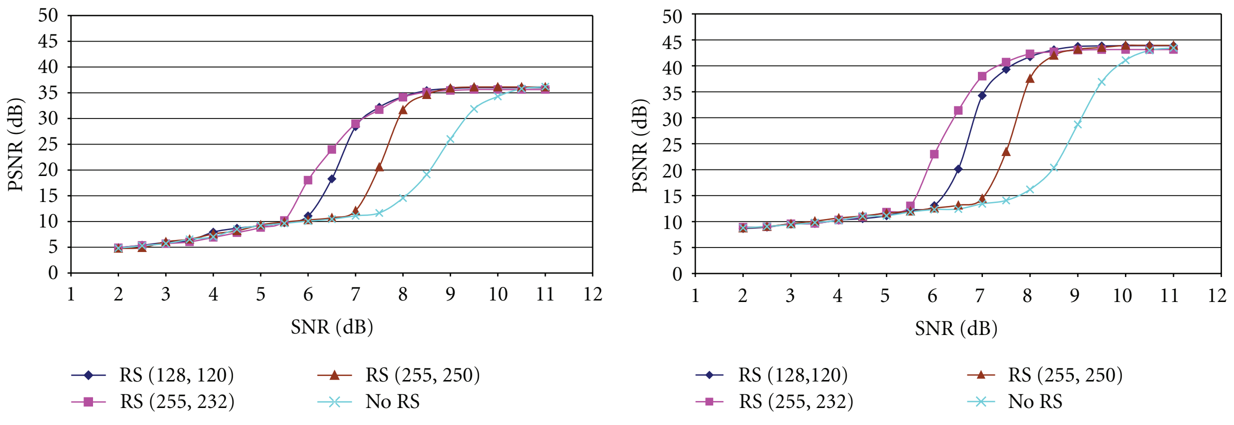


FIGURE 10: PSNR evolution for "Foreman" sequence in CIF resolution over a packet erasure channel for different protection levels with RS codes.

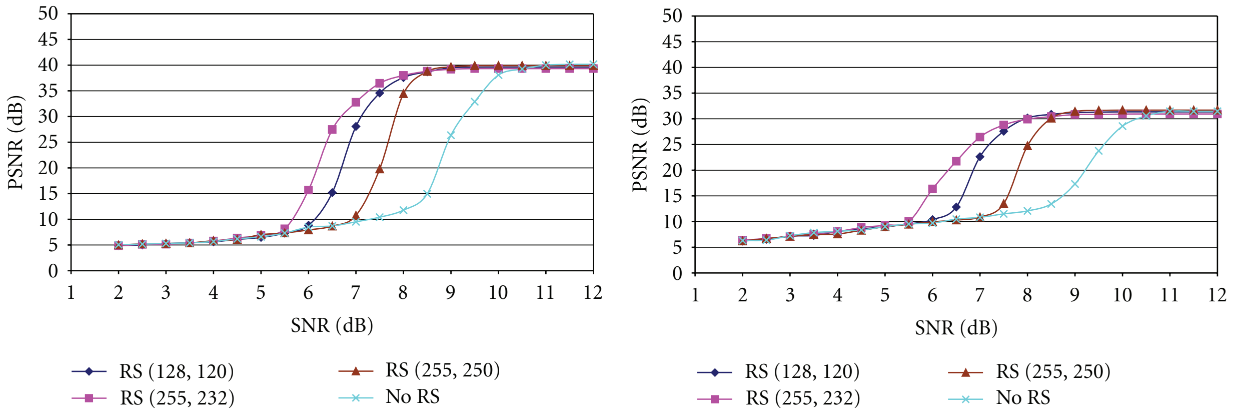
in our application, rather than the bit Error-Rate. Figure 9 presents the bit and frame Error-Rate performance curves of the obtained LDPC code over an Additive White Gaussian Noise (AWGN) channel, for a maximal of 80 iterations (in practice an average of 9 iterations are sufficient in the bottom of the waterfall region, i.e., after 2 dB).

When considering the different curves presented in Figure 10 to Figure 14, one clearly see appear different areas where the optimal protection level differs. In all cases, with either packet erasure channel (PEC) or binary symmetric channel, it can be observed that almost perfect channels will be greatly helped even by a very low redundancy.



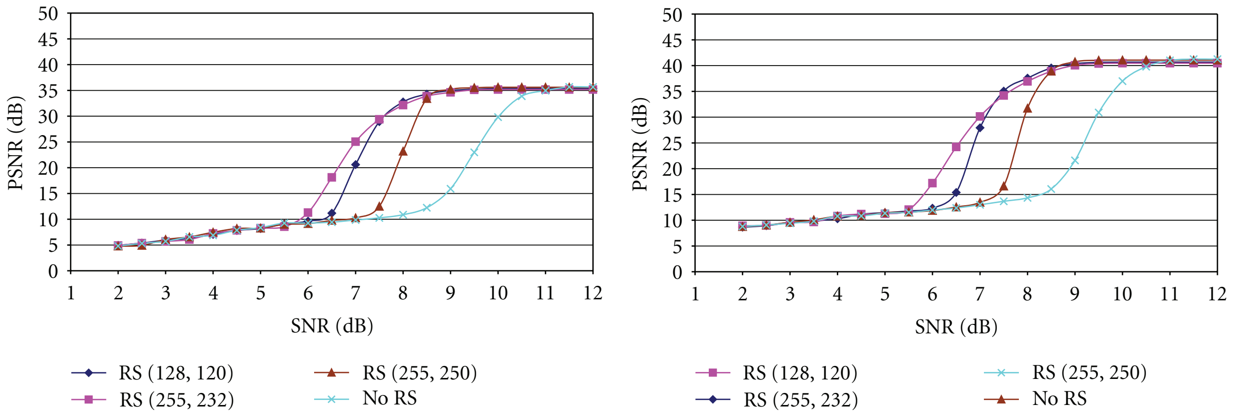
(a) (b)

Figure 11: PSNR evolution for "Foreman" and "Akyio" sequences in QCIF resolution over BSC channel for different protection levels with RS codes.



(a) (b)

Figure 12: PSNR evolution for "HallMonitor" and "Mobile Calendar" sequences in QCIF resolution over BSC channel for different protection levels with RS codes.



(a) (b)

Figure 13: PSNR evolution for "Foreman" and "Akyio" sequences in CIF resolutions over BSC channel for different protection levels with RS codes.

Typically, over the PEC channel, introducing protection with an RS(255,232) code allows to combat easily up to 5% loss rate, at almost no degradation of the original image.

Similarly, over the BSC channels, we see that an RS(255,250) code, which introduces less than 5% of redundancy already allows to gain noticeably. However, it is interesting to note that pursuing always greater redundancy level is not always the best choice: typically the performance obtained with the RS(128,120) code will tend to be more interesting than the one obtained with the RS(232,255) over the BSC channel when one considered the actual region of interest of PSNR, namely for values above 25 or 30 dB. This necessary balance can be explained by two reasons: first, as the global throughput is kept constant, using an higher redundancy rate implies to compress more the original data, and consequently to degrade the maximal quality attainable. This explains why at very high signal-to-noise ratio (SNR) the unprotected curve performs the best. The second reason is less obvious: simulations have shown that for lower SNR, the strong PSNR degradation comes from the fact that redundancy NAL units header is more often corrupted (corruption detected by the header checksum), leading to the impossibility to use the redundancy information. For that reason, increasing exaggeratedly the redundancy will not help in the proposed framework.

This issue disappears in the case where the NAL unit headers are protected, as could for instance be achieved when using length variable transport checksums (with UDP-Lite protocol for instance) with an RTP packetization taking only a single NAL unit per RTP packet. Figure 15 illustrates this effect, and shows that when the redundancy NAL unit is not corrupted, the correct decoding occurs even in presence of a much more degraded channel (up to 6 dB earlier in terms of channel SNR). In such a case, it becomes interesting to envisage error-correction codes operation with protection rate greater than the Reed-Solomon traditional range, for instance LDPC codes which are known to perform well for correction rates as low as 1/2 or 1/3. Figure 15(b) shows results obtained with a LDPC redundancy NAL units using the aforementioned (128,64) irregular LDPC code, showing an improvement due to the better performance of the LDPC of about 1.5 dB SNR when compared to the usage of an (128,64) RS code.

In all cases, one see appear different areas in which the optimal level of protection varies. When having a feedback link able to transmit the channel state quality at the transmission side, simple rules could be applied to allow an optimisation of the approach. For instance, QCIF sequences over a BSC channel could follow this simple rule

(i) for BER $< 5 \cdot 10^{-5}$ use RS (255,250) FEC code,

(ii) for BER $> 5 \cdot 10^{-5}$ use RS (128,120) FEC code.

This very simple unequal error protection approach with time could also be broadened by taking into account the type of slices present in the matrix: more bandwidth could be provided for one matrix containing high-importance frames, while another matrix with less important frames could be more compressed or less protected to recover the previously over-used bitrate. Interestingly, it should be noted that the capability to change the FEC protection in time is not limited to change the coding rate: the decoding process detailed in Section 3.2 making sure to check for each new redundancy NAL unit the used code (with the used code index) and its parameters, it is also possible to change in the middle of a sequence the type of protection, typically going from a low protection level with Reed-Solomon codes to a higher protection level with LDPC codes.

Finally, the results presented in Figure 16 allow to compare the performance obtained with 1/1 and N/M configurations, again in the case of a BSC channel. One finds that a noticeable gap is observed in favor to N/M approach when error probability is larger than 10^{-4}. In practice, it is between 10^{-4} and 6.10^{-3} to 10^{-2} that the largest additional gain is observed with the N/M method, ranging from 2 to 7 dB over the BSC, which justifies the interest of the N/M approach.

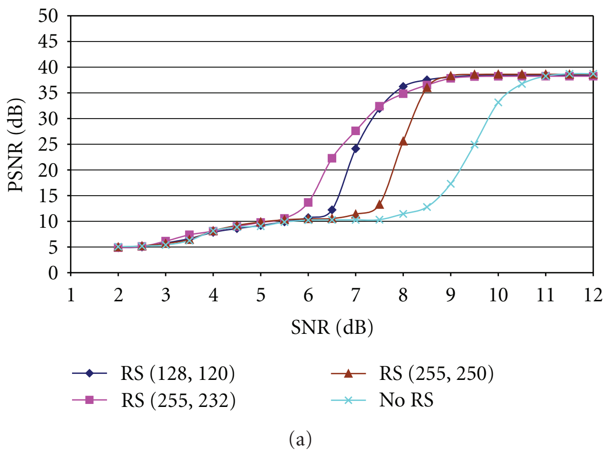
4.4. H.264/SVC Tests. As mentioned in Section 2.2, one of the interest of the proposed approach is its validity for the scalable extension of the H.264 standard [2].

To illustrate this, and demonstrate the interest of our protection embedding approach, we have tested our system with the "Foreman" sequence encoded in SVC format with three layers corresponding to QCIF, 15 Hz for the base layer, CIF, 15 Hz with additional enhancement layer 1 and CIF, 30 Hz with additional enhancement layer 2. Due to the very low resistance to errors of the current SVC verification model (called JSVM), we have chosen to report in Table 2 the decoding success probability of the complete stream (full resolution) in three configurations: without Reed-Solomon protection (here using an RS(128,120) code), with Reed-Solomon protection as detailed in Section 3.1 and finally with Reed-Solomon protection with redundancy NAL unit header uncorrupted. One observes with the Reed-Solomon code protection a very good decoding rate up to a BER of 10^{-5}, to be compared with a necessity of almost perfect channel when the scalable decoder is used alone. When the redundancy NAL unit header is protected, one can go up to bit Error-Rates of 10^{-4} with 71% chances of decoding success, with is becoming really interesting over error-prone channels.

5. Conclusions

This paper proposes a backward compatible mechanism to embed error protection inside H.264 (AVC or SVC) streams. The introduced solution relies on the insertion of supplementary network abstraction layer units that carry redundancy information generated by a systematic error-correction code. The proposed syntax for these supplementary NAL units is presented, together with tests results carried out with different error correcting codes (Reed-Solomon or LDPC codes) for H.264 AVC and H.264 SVC video streams, that show a noticeable performance gain for the video decoder over lossy or erroneous channels.

The possible use of this approach to realise an unequal error protection over time-varying channels has also been



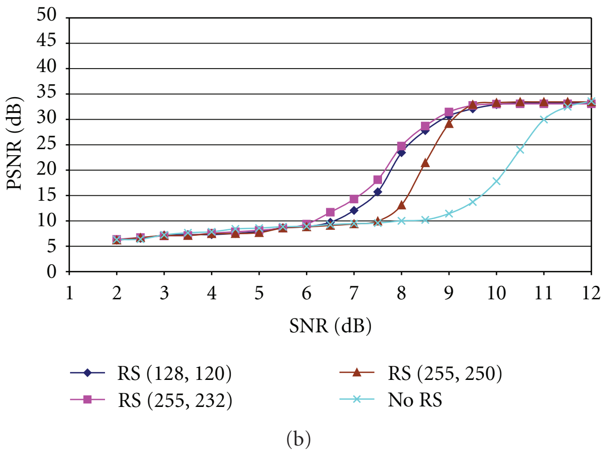
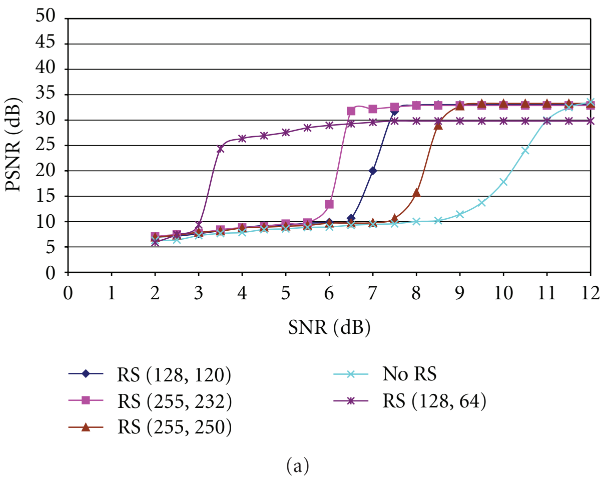


Figure 14: PSNR evolution for "HallMonitor" and "Mobile Calendar" sequences in CIF resolution over BSC channel for different protection levels with RS codes.



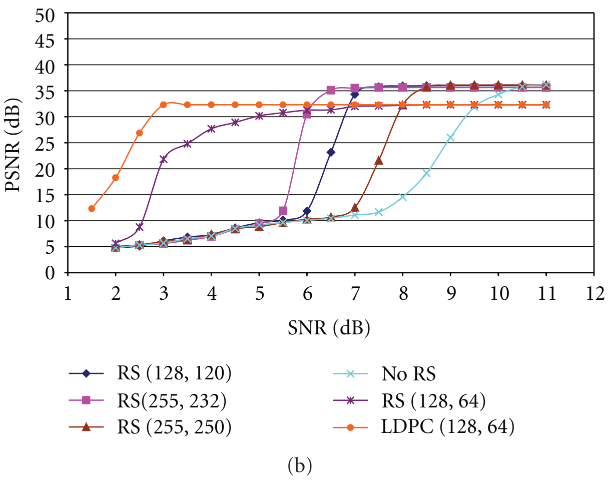
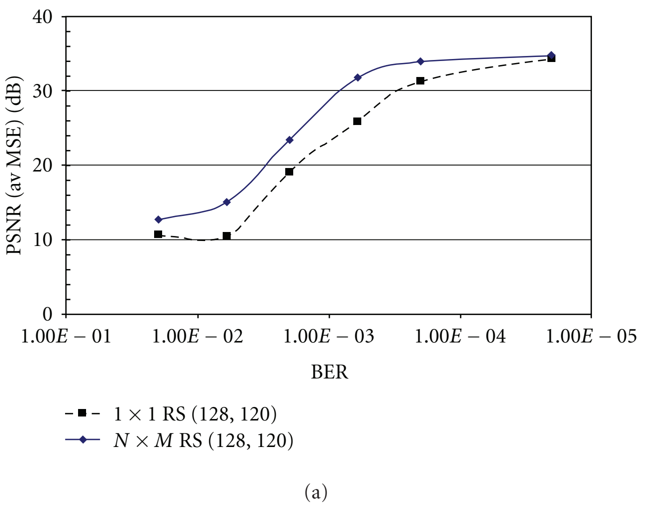


Figure 15: PSNR evolution for "Mobile Calendar" sequence in CIF resolution and "Foreman" sequence in QCIF resolution over BSC channel for different protection levels (RS or LDPC codes) and redundancy NAL unit headers protected.



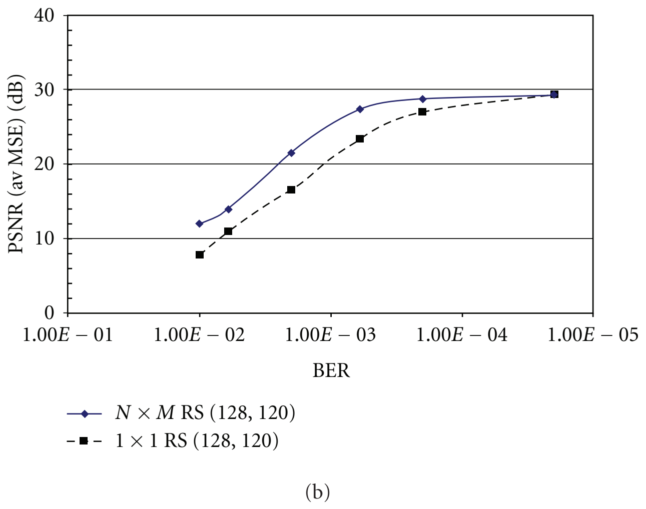


Figure 16: Comparing 1/1 and N/M redundancy approach performance for "Foreman" and "Mobile" QCIF sequence with RS(128,120) protection, BSC channel.

pointed out. Further investigations on this point and more generally the usage of this standard compatible FEC feature in the context of adaptive schemes are foreseen.

Acknowledgments

This paper was achieved with support partly of the French Industry Department through the POLE SYSTEM@TIC POSEIDON Project (no. 082930470) and partly of the European Community's Seventh Framework Programme through Grant agreement ICT OPTIMIX (no. INFSO-ICT-214625). C. Lamy-Bergot and B. Gadat want to gratefully acknowledge the support of their colleague Erwann Renan for his contribution of an efficient SVC codec for the simulations. The contributions of Karim Saada and Elodie Bounmy who participated to the development of the program during their internships at THALES Land and Joint Systems, EDS/SPM, France are also gratefully acknowledged.

References

[1] ITU-T Rec. H.264 — ISO/IEC 14496-10, 2003.

[2] ITU-T Rec. H.264 — ISO/IEC 14496-10 annex G, 2009.

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

[4] L.-W. Kang and J.-J. Leou, "An error resilient coding scheme for H.264 video transmission based on data embedding," vol. 3, pp. 257–260.

[5] D. Kim, S. Yang, and J. Jeong, "A new temporal error concealment method for H.264 using adaptive block sizes," in *Proceedings of the IEEE International Conference on Image Processing (ICIP '05)*, vol. 3, pp. 928–931, September 2005.

[6] T. Keranen, J. Vehkapera, and J. Peltola, "Error concealment for SVC utilizing spatial enhancement information," in *Proceedings of the 4th International Mobile Multimedia Communications Conference (MobiMedia '08)*, Oulu, Finland, July 2008.

[7] M. M. Ghandi, B. Barmada, E. V. Jones, and M. Ghanbari, "Wireless video transmission using feedback-controlled adaptive H.264 source and channel coding," *IET Communications*, vol. 3, no. 2, pp. 172–184, 2009.

[8] D. J. C. Costello Jr., J. Hagenauer, H. Imai, and S. B. Wicker, "Applications of error-control coding," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2531–2560, 1998.

[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] M. Watson, "Forward Error Correction (FEC) Framework," draft-ietf-fecframe-framework-07 (work in progress: expires Sept. 2010), March 2010.

[11] IETF RFC 5510, "Reed-solomon forward error correction (FEC) schemes," J. Lacan, V. Roca, J. Peltotalo, and S. Peltotalo, Eds., 2009.

[12] V. Sgardoni, M. Sarafianou, P. Ferré, A. Nix, and D. Bull, "Robust video broadcasting over 802.11a/g in time-correlated fading channels," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 1, pp. 69–76, 2009.

[13] S. Ahmad, R. Hamzaoui, and M. Al-Akaidi, "Adaptive unicast video streaming with rateless codes and feedback," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 2, pp. 275–285, 2010.

[14] M. G. Martini, M. Mazzotti, C. Lamy-Bergot, J. Huusko, and P. Amon, "Content adaptive network aware joint optimization of wireless video transmission," *IEEE Communications Magazine*, vol. 45, no. 1, pp. 84–90, 2007.

[15] F. G. MacWilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes: Part 1*, North-Holland Publishing Company, New York, NY, USA, 1977.

[16] R. G. Gallager, *Low-Density Parity-Check Codes*, MIT Press, Cambridge, Mass, USA, 1963.

[17] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 386–398, 2005.

[18] S. Valente, C. Dufour, F. Grolière, and D. Snook, "An efficient error concealment implementation for MPEG-4 video streams," *IEEE Transactions in Consumer Electronics*, vol. 47, no. 3, pp. 568–578, 2001.

[19] D. Agrafiotis, D. R. Bull, and N. Canagarajah, "Optimized temporal error concealment through performance evaluation of multiple concealment features," in *Proceedings of the International Conference on Consumer Electronics (ICCE '06)*, pp. 211–212, January 2006.