# Analysis of Packet-Level Forward Error Correction for Video Transmission

Matteo Mazzotti*, Enrico Paolini*, Marco Chiani*, Benjamin Gadat†, Cyril Bergeron†, Roberta Fracchia†
*DEIS/WiLAB, University of Bologna, via Venezia 52, 47521 Cesena (FC), Italy
†THALES Communications, 160 boulevard de Valmy, 92704 Colombes Cedex, France
Email: {mazzotti.matteo,e.paolini,marco.chiani}@unibo.it, {Benjamin.Gadat,Cyril.Bergeron,Roberta.Fracchia}@fr.thalesgroup.com

***Abstract*—In this paper, packet-level coding is considered in the framework of H.264/AVC video transmission. Two distinct solutions are proposed and compared in different realistic communication scenarios. The first is based on classical Reed-Solomon (RS) codes applied at the RTP layer, while the second on modern LDPC codes implemented at the UDP-Lite layer. An end-to-end Quality of Experience (QoE) evaluation is presented, in terms of achieved peak signal-to-noise power ratio (PSNR). Our numerical results show that, in low-latency video applications across communication channels introducing errors and erasures, the adoption of a packet-level coding scheme becomes essential to guarantee a satisfactory quality. The solution based on LDPC codes exhibits better performances in presence of severe packet loss rates.**

## I. Introduction

Packet-level codes have recently gained an increasing attention, especially in delay-sensitive and broadcast/multicast multimedia communication systems [1], [2], [3]. In video transmissions, packet-level codes can be implemented at application or transport layer, with real time protocol (RTP) or UDP/UDP-Lite protocols. In this paper, we evaluate and compare the performance of two packet-level coding schemes, in the context of H.264/AVC video transmissions. The first scheme consists in adopting systematic Reed-Solomon (RS) codes at RTP video fragmentation. The approach followed in the second scheme is based on systematic low-density parity-check (LDPC) codes with efficient maximum likelihood (ML) decoding [4], [5], similarly to what recently included in the UMTS standard [6].

The principles of packet-level coding are reviewed in Sec. II. In Sec. III, the scheme based on RS codes applied at RTP level is described, while Sec. IV is devoted to detail the LDPC-based solution, implemented at the transport layer. Some numerical results are presented in Sec. V, based on the complete simulation platform developed within the European FP7-ICT OPTIMIX project [7].

## II. Packet-Level Codes at RTP/Transport Layer

Both packet-level coding schemes analyzed in this paper are aimed to improve H.264/AVC video transmission. The first scheme operates at RTP layer, while the second one at transport (UDP-Lite) layer.

In packet-level coding, redundant packets (also called *repair packets*) are generated at the transmitter side, starting from a set of data packets (also called *source packets*) incoming from the upper layers. This redundancy is aimed at enabling the recovery of packet erasures and the correction of bit errors at the receiver side. In general, the encoder updates the source packets with additional data containing configuration information necessary for the decoding process, and forwards them immediately to the lower layers to introduce no significant extra-delay.

The encoding process starts when either a sufficient amount of data is available, or a maximum tolerable delay has been reached. The first step consists in filling an encoding table, also called the *source block*, with the source packets. The source block is composed of $n = k + m$ rows, each of $T$ bytes, indexed from 1 to $n$. Each such row is called a *symbol*. The source packets can be inserted only in the first $k$ rows of the table, in a progressive order. Hence, the first $k$ rows of the table are named source symbols. In the second step of encoding, $m = n - k$ repair symbols are generated from the $k$ source symbols, and transmitted as the payload of repair packets. The length and number of repair packets formed from the $m$ repair symbols are design parameters changing with the specific implementation.

Dually, at the decoder, the received source and repair packets are used to fill an identical table, called *decoding source block*. Each received packet is inserted in the same position occupied in the corresponding source block, exploiting proper control information included the packet. Packet losses may be experienced at the receiver, due to erasures along the IPv6 network, errors in the RTP/UDP/IP/DLL headers caused by the transmission across the radio channel, or selective drops possibly operated to adapt the video stream to network congestion conditions. In addition to packet losses, partially corrupted (source or repair) packets, i.e. packets whose payload is recognized as corrupted but without errors in the header, may be received by the packet-level decoder. To this purpose, it is necessary to adopt a transport protocol including partial checksum features, such as UDP-Lite [8], along with a data link layer supporting a specific CRC code to protect the frame header.[1]. Partially corrupted video packets may be treated by the packet-level decoder either as erasures, in which case they

[1]For instance, this mechanism is supported by the WiMAX MAC layer, through the header check sequence

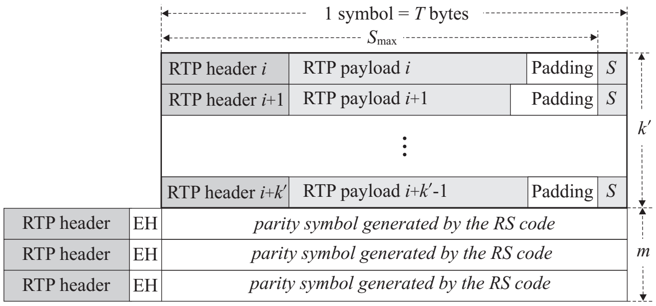


Fig. 1. Structure of the source block after encoding with the RS packet coding scheme. Proper RTP headers and extra-header fields are prepended to the parity symbols to generate the RTP parity packets.

are not filled into the decoding source block, or as information affected by some (unknown) error pattern. Decoding consists in recovering the missing $T$-byte symbols from erasures and, if partially corrupted packets are not treated as erasures and are filled into the table, correcting the errors affecting the corresponding $T$-byte symbols.

At the end of the decoding process, $T$-byte symbols still in error may be present. The corresponding partially corrupted video packets may be forwarded to the application layer as well, since robust decoders are usually able to exploit them to improve the end-to-end video quality. This general framework is detailed in the following for the two considered packet-level coding techniques.

## III. RS-Based RTP FEC approach

An RS code of length $n_{\mathrm{RS}}$ and dimension $k_{\mathrm{RS}}$ is a linear block code capable to correct any combination of $E$ errors and $L$ erasures as long as the relation $2E + L \leq n_{\mathrm{RS}} - k_{\mathrm{RS}}$ is satisfied. These codes are thus especially well suited to protect the transmission from bursts of errors. Moreover, as they are systematic, they allow to introduce redundancy without affecting the information symbols: a solution based on RS codes is thus fully transparent for users unaware of the forward error correction (FEC) feature.

The number $T$ of bytes in each row is set to the maximum RTP packet size $S_{\max}$ plus 2 additional bytes indicating the packet size, as depicted in Fig. 1. The rationale behind this choice is as follows. RTP packets have a variable size, which can assume a value between 13 bytes (i.e., an header size of 12 bytes and a payload of 1 byte) and $S_{\max}$. At the receiver side, in case of losses, the information about the packet size is needed for the recovery process: we propose to append this extra information to the RTP source packets while inserting them in the source block. The packet size $S$, virtually inserted in the matrix and thus considered in the generation of the RTP repair packets, is not really transmitted to the receiver: at the receiver side, the payload length can be recovered from the RTP repair packets in case of losses, since it has been included in the computation of the protection.

The employed RS code is built on the Galois field GF($2^3$) and its length is $n_{\mathrm{RS}} = 255$ symbols. The packet-level encoder builds a source block with $k$ rows, where each row is filled with a specific RTP source packet. If an RTP packet does not fill a row entirely, padding is added. The $k$ RTP source packets, obtained fragmenting upper layer video frames as specified by the standard [9], are transmitted to the receiver followed by $m$ RTP repair packets. Each repair packet is generated by $T$ RS encoders working in parallel, each one processing a specific column of the source block.

The dimension $k$, i.e. the number of source packets in the matrix, is a parameter of the system. However, it may happen that, in case of an H.264 video packetization, the $k$th packet is in the middle of a network abstraction layer (NAL). In order to avoid the split of a NAL into two different source blocks, we decided to insert in the matrix additional RTP packets, up to the completion of the NAL transmission. It follows that a number $k' = k + K$ of RTP source packets are inserted in the matrix, with $K \geq 0$. From a practical point of view, the coding process is based on a low code-rate RS "mother" code (e.g., an RS$(255, 128)$ code) and the target $k'$ and $n$ are matched by shortening and puncturing the mother code.

The $m = n - k$ RTP repair packets are characterized by a new payload type and an "extra-header". The use of a new payload type (e.g., no. 99) assures the compliance to RTP receivers not supporting packet-level FEC: if RTP repair packets are received by a client not supporting the RTP FEC mode, the packets with the unknown payload type are simply discarded. An RTP FEC receiver would instead decode the repair RTP packets and correct the possible errors or losses.

The RTP extra-header of the repair packets is composed of four bytes, namely:

- a sequence number SBN (2 bytes), identifying the first packet of the source block;
- the value of $k'$ (1 byte, for a maximum number of rows $n_{\mathrm{RS}} = 255$);
- an extra-header error control field (1 byte). It has to be noticed that the sequence number and $k'$ have the same values for all the packets in the same source block: in case of errors, these values can be obtained from the other correct packets of the matrix. One byte of protection is thus, in our opinion, enough to offer a good reliability with a minimum overhead.

## IV. LDPC-Based Packet Erasure Coding

As opposed to the previous scheme, where source and repair packets are RTP packets, in this implementation UDP-Lite source packets are processed by the packet-level encoder to generate UDP-Lite repair packets. This scheme is almost compliant to that recommended for the UMTS system multimedia broadcast/multicast service (MBMS), where Raptor codes are adopted instead of LDPC codes, and can be in principle applied also to non-RTP application streams.

### A. Co-Decoding Procedures

For a given $(n_{\mathrm{LDPC}}, k_{\mathrm{LDPC}})$ LDPC code, the source block is composed of $n_{\mathrm{LDPC}}$ rows, each of $T$ bytes. In this case, $T$ is a design parameter, independent of the maximum length

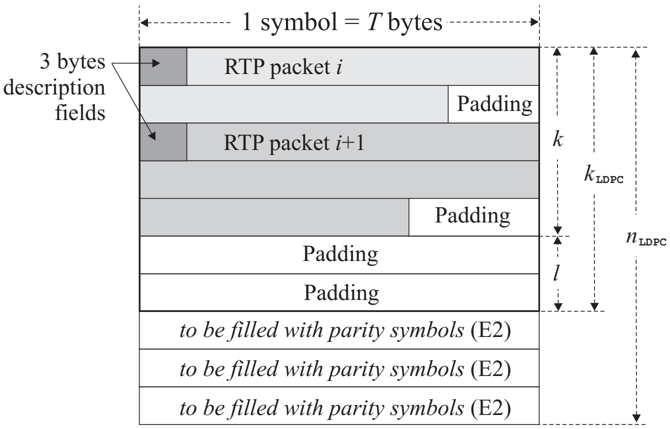


Fig. 2. Structure of the source block before encoding in the LDPC-based packet level case. This matrix will be then encoded through the steps E1-E3. In particular, during E2, $n_{\text{LDPC}} - k_{\text{LDPC}}$ parity symbols will be generated and inserted in the last rows of the source block. In general, some parity symbols may be punctured to match the desired code rate $R_c$.

of source packets (typically $1 \leq T \leq 16$). The first $k_{\text{LDPC}}$ rows of the source block are filled with the RTP packets constituting the payload of UDP-Lite source packets. Proper description fields (3 bytes) are pre-pended to each of the inserted packets, to indicate the transport flow ID (associated to a UDP destination port and an IP destination address – 1 byte) and the RTP packet length. A zero padding completes the last row occupied by the RTP packet.

After the first $k_{\text{LDPC}}$ source block rows have been filled with RTP packets, encoding is performed to generate $n_{\text{LDPC}} - k_{\text{LDPC}}$ repair symbols. A certain number of UDP-Lite repair packets is then formed to match the objective code rate $R_c$, where the payload of each repair packet is composed of several repair symbols. Repair packets are sent to specific destination ports of the transport layer, devoted to this service.[2] If no additional source packets are available, encoding is imposed by a time trigger. In this case, the $l$ rows of the source block out of the first $k_{\text{LDPC}}$ rows which have not yet been filled, are set to all-zero. The overall process is equivalent to use an $(n, k)$ code obtained by, first, expurgating the mother LDPC code and, then, puncturing it (i.e., $k = k_{\text{LDPC}} - l$, with $l \geq 0$, and $n = n_{\text{LDPC}} - p$, where the number of punctured symbols $p$ satisfies $p \geq l$). The information about which rows have been filled with source packets and which ones have been filled with padding are made available at the receiver, as explained later.

The structure of the source block before encoding is depicted in Fig. 2. To mitigate the effect of erasure bursts caused by UDP-Lite packet losses, an interleaving process has been designed with the additional goal to transmit source and repair symbols in the payload of distinct packet types (source and repair packets, respectively). More in detail, encoding is

[2]The same approach can be used also with connection-oriented transport protocols, identifying a connection based on the source and destination port numbers and IP addresses. In this case, two different connections (for source and repair packets respectively) have to be opened.

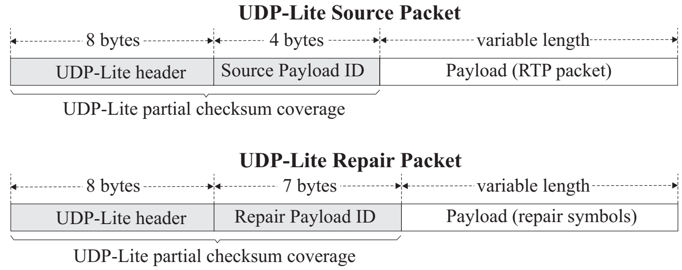


Fig. 3. Structure of UDP-Lite source and parity packets.

performed according to the following steps:

E1. Pseudo-random interleaving of the systematic part of the source block (first $k_{\text{LDPC}}$ rows). The pseudo-random generator is initialized with the source block number, which is known both at the transmitter and at the receiver.
E2. Generation of the $n_{\text{LDPC}} - k_{\text{LDPC}}$ repair symbols.
E3. Pseudo-random interleaving of the redundant part of the source block (last $n_{\text{LDPC}} - k_{\text{LDPC}}$ rows).

At the decoder, the payload of the received source and repair packets is inserted in the decoding source block coherently with its position in the encoding source block, which is known due to specific side information (described in Sec. IV-B) included in the packets. Clearly, the same 3 bytes field used at the transmitter side is prepended to each source payload before filling it into the matrix. In this packet-level coding implementation, partially corrupted source and repair packets are treated as erasures. Note that partially corrupted *source* packets are not discarded.[3] If the corresponding RTP packet is recovered by the erasure decoding process, then its recovered version is forwarded to the H.264 video decoder. Otherwise, the partially corrupted version, not directly exploited by the decoder, is forwarded to the upper layer. On the contrary, if received repair packets contain bit errors, they are immediately discarded as they are not useful for the video decoder. Decoding is performed as follows:

D1. Pseudo-random interleaving of the systematic part of the decoding source block (first $k_{\text{LDPC}}$ rows), according to the source block number.
D2. Deinterleaving of the redundant part of the decoding source block (last $n_{\text{LDPC}} - k_{\text{LDPC}}$ rows).
D3. Iterative or ML LDPC erasure decoding.
D4. Deinterleaving of the systematic part of the decoding source block.

At the end of decoding, the RTP source packets are extracted based on the corresponding 3 bytes description fields.

### B. Packets Format

The structure of a source packet, depicted in Fig. 3, consists of:

- UDP-Lite header;

[3]The presence of bit errors in the payload of the UDP-Lite packets can be signalled by the DLL as cross-layer information.

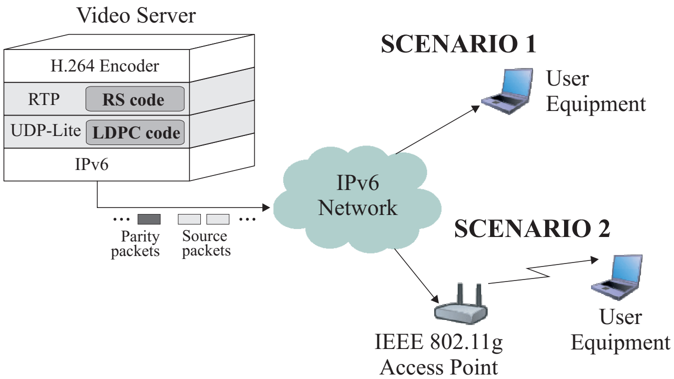


Fig. 4. Communication scenarios considered in our analysis. In scenario 1 the user is connected to the video server through wired links and packet losses may happen within the IPv6 network. In scenario 2 an additional wireless link may introduce further packet erasures as well as some bit-errors in the payload of the transmitted packets.

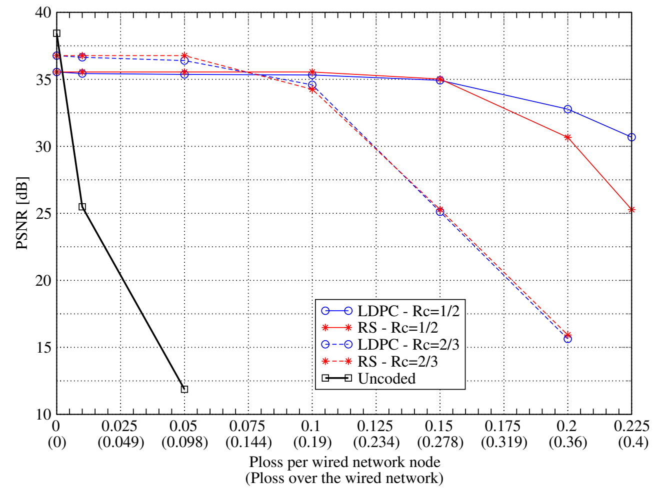


Fig. 5. PSNR versus packet loss probability in scenario 1, including only wired links. Packet losses are introduced in the IPv6 network.

- Source Payload ID (4 bytes), including the source block number (2 bytes) and the encoding symbol ID (ESI), i.e. the index of the first row in the source block occupied by the packet (2 bytes);
- Payload (RTP packet).

On the other hand, a repair packet is composed of the following parts, as depicted in Fig. 3:

- UDP-Lite header;
- Repair Payload ID (7 bytes), including the source block number (2 bytes), the ESI[4] (2 bytes), the number $k = k_{\text{LDPC}} - l$ of used systematic rows (2 bytes), and the LDPC code ID (1 byte) identifying the specific LDPC code and its code rate $R_c$.
- Payload (repair symbols).

For both packets types, the UDP-Lite header and the payload ID are particularly important fields, since they contain information necessary to perform the decoding process. For instance, the UDP-Lite destination ports specified in the packet header allow to discriminate between source and repair packets, while the payload ID contains the ESI information, which is necessary to insert the packet within the source block. For this reason, we decided to protect the first two parts of each packet through the partial checksum coverage mecanism supported by UDP-Lite [8]: if bit errors affect the UDP-Lite header or the payload ID, at the receiver side the packet is immediately discarded and considered as an erasure during the decoding.

## V. Simulation Results

To compare the two proposed packet-level schemes, we evaluated their performance through multiple Monte Carlo simulations, based on the powerful software framework developed within the ICT European project OPTIMIX. This platform realistically models all the communication blocks, from application to physical layer. For details about this simulator, we refer the reader to [7].

[4]In this case, the ESI represents the row index offset with respect to $k_{\text{LDPC}}$.

Two different communication scenarios have been considered, as depicted in Fig. 4. Users are connected to the video server through both multiple wired links (Internet) and a wireless hop between the network and the terminal (e.g., a WLAN). In scenario 1, the receiver has to deal only with packet losses, while in scenario 2 the additional wireless channel may introduce further packet losses and bit errors within the packet payloads. In the latter scenario, MAC and transport layer protocols discard the packets with corrupted headers and trasfer to the upper layers the packets having bit errors in the payload only. We assume that retransmissions are not allowed at any level, due to strict delay and bandwidth constraints.

In our simulations, the IPv6 wired network is composed by $N = 2$ nodes, each introducing statistically independent packet losses with probability $P_{\text{loss}}$. For any $N$, the overall packet loss probability across the network is given by $\hat{P}_{\text{loss}} = 1-(1-P_{\text{loss}})^N$. The wireless physical layer has been modelled according to the IEEE 802.11g standard, based on OFDM modulation with $48$ data subcarriers. A QPSK constellation is adopted on each subcarrier and a convolutional code with rate $1/2$ is employed at the physical layer to protect the data. A Rayleigh block fading channel models the wireless link, with a coherence time of $0.1$ s and a coherence bandwidth of approximately 2.5 MHz. We simulated the transmission of an H.264/AVC encoded "Foreman" sequence of $10$ s, in CIF format and with a frame rate of 30 fps. Each Group Of Pictures (GOP) is composed of 30 frames. We present in the following the impact of losses and bit errors on the received video quality measured in terms of peak signal-to-noise power ratio (PSNR) [10].

In Fig. 5, the PSNR performance of the two packet-level coding schemes is depicted in the case of scenario 1. For the sake of fairness, the size (in byte) of the systematic part of the souce block is the same for both solutions ($\sim$ 20 kB). More specifically, in the scheme based on RS codes each symbol

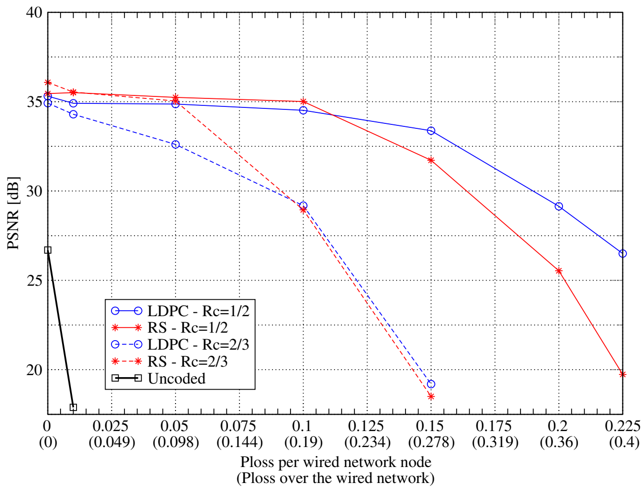


Fig. 6. PSNR versus network loss probability in scenario 2, including wired links (IPv6 network) and an IEEE 802.11g hop. Besides packet losses, bit errors are introduced in the wireless link, characterized by $E_s/N_0 = 28$dB. Packets with corrupted payloads are transferred to the application layer.

is set to $T = 1024$ bytes and $k = 20$, while in the LDPC-based scheme $T = 5$ bytes and $k_{\mathrm{LDPC}} = 4096$. Two different code rates, $R_c = 1/2$ and $R_c = 2/3$, are adopted for both schemes. The ML decoder based on the *maximum column weigth* pivoting algorithm in [11] is adopted for LDPC codes. The PSNR obtained with an uncoded transmission has been reported as a reference. Different video compression levels have been considered based on the applied code rate, in order to achieve the same average throughtput after the RTP/UDP-Lite encapsulation ($\sim 840$ kbps). Each point in the graph was obtained averaging the PSNR results of 20 simulation runs with different noise seeds.

In scenario 1, when $P_{\mathrm{loss}} = 0$, the PSNR is higher in the uncoded case. Indeed, the 840 kbps of bit rate are entirely devoted to the video. When packet-oriented protection is applied, the bit rate of the video is reduced by an amount equivalent to the introduced redundancy: in this case, the transmitted video has a lower definition. Since no losses affect the transmission, no benefit is achieved by packet-level coding. Similarly, a low video compression coupled with a higher code rate ($R_c = 2/3$) is preferable with a low $P_{\mathrm{loss}}$, while increasing $P_{\mathrm{loss}}$, a lower code rate ($R_c = 1/2$) applied to a highly compressed video achieves a better PSNR, since the lower video definition is counterbalanced by the recovery of lost packets. As it can be noticed, applying the packet-oriented protection outperforms unprotected transmissions even for very low values of $P_{\mathrm{loss}}$. Comparing the two solutions, we note that the performances for $R_c = 2/3$ are very close. On the contrary, the major length of the code favors the LDPC over the RS with high values of $P_{\mathrm{loss}}$.

In Fig. 6 we report some results obtained in scenario 2, considering an IEEE 802.11g radio access link with an average $E_s/N_0 = 28$ dB, where $E_s$ is the energy per OFDM symbol and $N_0$ is the one-sided noise power spectral density. Besides considerations similar to what already reported for Fig. 5, we note that, in this case, the RS solution allows to reach PSNR values slightly higher than the LDPC code when the packet loss probability is low ($\hat{P}_{\mathrm{loss}} < 0.2$). This is basically due to the bit error correcting capabilities of the RS solution, allowing to recover packets which would be treated as erasures in the LDPC-based scheme.

## VI. Conclusion

We described two packet-level coding solutions based on RS and LDPC codes, respectively, for H.264/AVC video applications. Their performance in terms of PSNR vs packet loss rate was evaluated through several numerical simulations in two different communication scenarios, comprising both wired and wireless links. Both schemes are capable to heavily improve the final QoE: in particular, the bit-level correction supported by the RS solution provides some advantages over the LDPC code at low packet loss rates. On the contrary, increasing the erasure rate, the powerful LDPC-based scheme determines better video qualities.

## Acknowledgment

This work has been carried thanks to INFSO-ICT-214625 OPTIMIX project, partially funded by the European Commission within the EU 7th Framework Programme and Information Society Technologies.

## References

[1] J. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 8, pp. 1528–1540, Aug. 2002.

[2] A. Li, "RTP Payload Format for Generic Forward Error Correction," RFC 5109 (Proposed Standard), Internet Engineering Task Force, Dec. 2007.

[3] J. Neckebroek, M. Moeneclaey, and E. Magli, "Comparison of Reed-Solomon and Raptor codes for the protection of video on-demand on the erasure channel," in *Proc. of 2010 Int. Symp. on Information Theory and its Applications*, Oct. 2010, pp. 856–860.

[4] D. Burshtein and G. Miller, "An efficient maximum likelihood decoding of LDPC codes over the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2837–2844, Nov. 2004.

[5] E. Paolini, G. Liva, B. Matuz, and M. Chiani, "Generalized IRA erasure correcting codes for hybrid iterative / maximum likelihood decoding," *IEEE Commun. Lett.*, vol. 12, no. 6, Jun. 2008.

[6] 3GPP TS 26.346 V8.0.0, "Technical specification group services and system aspects; multimedia broadcast/multicast service; protocols and codecs," Sep. 2008.

[7] R. Fracchia, M. Mazzotti *et al.*, "Optimisation of multimedia over wireless IP links via X-layer design: an end-to-end transmission chain simulator," *Multimedia Tools and Applications* (Springer Netherlands), pp. 1–28, 2010.

[8] L.-A. Larzon, M. Degermark, S. Pink, L.-E. Jonsson, and G. Fairhurst, "The Lightweight User Datagram Protocol (UDP-Lite)," RFC 3828 (Proposed Standard), Internet Engineering Task Force, Jul. 2004.

[9] S. Wenger, M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer, "RTP Payload Format for H.264 Video," RFC 3984 (Proposed Standard), Internet Engineering Task Force, Feb. 2005.

[10] L. Hanzo, P. Cherriman, and J. Streit, *Video compression and communications: from basics to H.261, H.263, H.264, MPEG4 for DVB and HSDPA-style adaptive turbo-transceivers*. IEEE Press, 2007.

[11] G. Liva, B. Matuz, E. Paolini, and M. Chiani, "Pivoting algorithms for maximum likelihood decoding of LDPC codes over erasure channels," in *Proc. of 2009 IEEE Global Telecommunications Conf.*, Nov./Dec. 2009, pp. 1 –6.