Link-Level Partial Checksum for Real-Time Video Transmission over 802.11 Wireless Networks

Link–Level Partial Checksum for Real–Time Video Transmission over 802.11 Wireless Networks

E. Masala, M. Bottero and J.C. De Martin*
Dipartimento di Automatica e Informatica
*Istituto di Elettronica e di Ingegneria dell'Informazione e delle Telecomunicazioni — IEIIT–CNR
Politecnico di Torino, Italy
E-mail: [masala | mauro.bottero | demartin]@polito.it

***Abstract*— We present a technique to enhance the 802.11 link–layer effectiveness for video packets by changing the scope of the standard 802.11 error detection step. According to the proposed technique, the link–layer retransmits video packets only if errors are detected in the most sensitive bit class, instead of retransmitting all corrupted packets irrespective of errors number and position. A test implementation of the partial checksum technique for H.264 video transmission is presented, where the link–level checksum protects only the most important subset of the compressed video bitstream, as defined by the ITU–T H.264 standard. Simulation results —obtained using NS–2 and a channel model based on experimental error traces— show that the negative impact of errors on the less sensitive bits is counterbalanced by the lower number of discarded video packets, lower average delay and reduced network load. The proposed partial checksum technique provides consistent video quality improvements, up to 1–1.5 dB, with respect to the standard full–checksum technique as well as to other state–of–the–art techniques, such as header–only protection and unequal error protection.**

I. Introduction

In recent years the IEEE 802.11 wireless local networking (WLAN) standard [1] has gained widespread popularity as a method to allow wireless access to the Internet. From plain Internet access, typically used for applications such as email and web browsing, the focus is now shifting towards Multimedia applications over WLAN, including Wi-Fi telephony, streaming, videoconferencing, and wireless collaborative gaming.

However, several challenges need to be addressed to provide successful real-time multimedia applications over a network originally designed for generic data packet traffic and characterized by potentially high error rates. Multimedia data, in fact, are characterized by strictly bounded quality of service requirements in terms of packet losses, end-to-end delays and jitter.

Many schemes have been proposed to enhance the performance of multimedia communications over 802.11 wireless LANs. For instance, layered coding coupled with Unequal Error Protection (UEP) obtained by using different retry limits at the link level has recently been shown to deliver interesting results [2] [3] [4].

This work was supported in part by MIUR, Project FIRB-PRIMO, http://primo.ismb.it.

An important characteristic of multimedia traffic is that damaged packets are often preferable to outright lost packets. Multimedia compression algorithms, in fact, typically provide a certain degree of error resilience that can be exploited by the decoder to recover useful information even in partially corrupted packets. The current IEEE 802.11 Media Access Control (MAC) layer, however, prevents the forwarding of erroneous packets, irrespective of error number and position. Corrupted packets are discarded and the sender will retransmit the data until a given maximum retransmission limit (retry limit) is reached [1].

Recently, the UDP Lite transport protocol [5] has been employed to address the issue of partially corrupted packets in wireless multimedia communications. Video transmission using UDP Lite over a cellular access network has been studied in [6], showing that UDP Lite allows throughput improvements and end-to-end delay reductions which can benefit delay-sensitive applications that do not have stringent error requirements. The performance of the UDP Lite protocol is also evaluated in [7] by means of actual 802.11b experiments using different physical layer transmission speeds. An analytical byte-level channel model is built from the error traces. The impact of errors at the application level is considered for the case of video, showing that the quality degrades significantly increasing the physical transmission speed. In that work, however, the coverage of the UDP Lite checksum is limited to protocol headers, and the 802.11 MAC level error checking feature is completely disabled, hence no MAC level retransmissions are used.

In [8], a modified version of the UDP Lite protocol is proposed. It features a checksum for the packet header, and at the same time it provides an interface to forward all the information supplied by the CRC failures in link-layer frames to the application layer, to improve error location inside the packets. This protocol is combined with a UEP scheme applied to fixed-size link-layer frames in a 3G wireless scenario.

In [9] an architecture for multimedia transmission based on UDP Lite partial checksum coupled with congestion control is presented showing significant throughput improvements in a wired-wireless scenario. Others suggest to limit the UDP Lite partial checksum to the packet header [10], focusing on schemes that add redundancy at the data link level, and

allowing packets containing errors to be forwarded to the applications. Both works, however, do not measure the quality of the received multimedia streams.

These works, however, do not consider the possibility of a partial checksum at the *link* layer. Link-layer error detection is particularly attractive for moderate or high end-to-end delay scenarios, where end-to-end retransmission schemes are generally not applicable. Hop-based retransmission, in fact, is very fast, delivering "acceptable" packets with lower delay.

In this paper we propose to modify the IEEE 802.11 network link-layer to better support video communications allowing partially corrupted packets to be forwarded (and not discarded). A similar approach has already been explored for speech communication [11] [12]. A video transmission scheme is proposed, in which video packets are retransmitted by the MAC protocol only if errors are detected in the most sensitive subset of the compressed bistream, that —if corrupted— would introduce noticeable visual artifacts into the decompressed video sequence. Simulation results show that the negative effects of errors in the less sensitive bits are counterbalanced by the lower number of discarded video packets, lower average delay and reduced network load. Comparisons with the standard full-checksum technique as well as state-of-the-art techniques such as unequal error protection and header-only protection demonstrate that the proposed technique delivers consistent video quality improvements.

The paper is organized as follows. In Section II we briefly review the 802.11 wireless standard. Section III illustrates the advantages of the proposed partial checksum scheme, while Section IV presents an analytical description of its behavior. Section V describes an implementation of the partial checksum scheme for the specific case of H.264 video transmission. Section VI explains the experimental setup, including encoder configuration, wireless channel modeling and simulation scenario. Results, including comparisons with reference schemes, are presented in Section VII. Conclusions are drawn in Section VIII.

II. 802.11 Wireless Standard

In an 802.11 wireless LAN architecture, the fundamental access method used to support asynchronous data transfer on a best effort basis is the Distributed Coordination Function (DCF). It operates in a contention mode, requiring all stations to contend for each transmitted packet to gain access to the channel [13].

In the IEEE 802.11 MAC [1], each data-type frame consists of a MAC header, a variable length information frame body, and a Frame Check Sequence (FCS). Except for the wireless bridging case, the MAC protocol overhead is 28 bytes, composed of a 24-byte MAC header and a 4-byte trailer containing a checksum (FCS) of the MAC frame.

Upon frame reception the destination station compares the frame FCS with a new one computed over all the received MAC bits. Only if all the bits are correct the frame is positively acknowledged by sending an ACK frame back to the source station. When, after a network error, an ACK is not received, the source station contends again for the channel to transmit the unacknowledged packet and, in case of further error, retries until a given maximum retry limit is reached [1]. Therefore, the time needed to successfully send a MAC frame can be quite large if compared with the physical layer transmission time of a single MAC frame. Note, in fact, that a new channel contention phase is needed for each retransmission, and the size of the Contention Window (CW), i.e. the time interval in which the station randomly selects the instant of the next transmission attempt, doubles after each unsuccessful retransmission.

III. Partial Checksum for Wireless Multimedia

Wireless networks, unlike wired networks, are prone to bit errors. However, as long as errors affect the less percetually important data bits, modern multimedia applications can often deal with corrupted frames better than with lost ones by means of error resilient coder design and error concealment techniques. But if sensitive information is corrupted, such as header data, the video quality may degrade considerably, and some decoders could crash.

Consider, for instance, a receiver station that detects corruption in a packet. On the one hand, the protocol stack could decide to simply discard it. In this case, the video decoder would trigger an error concealment routine that handles the missing packet. On the other hand, despite the errors, the packet could be forwarded to the video decoding engine that can at least partially decode the packet, thus minimizing the distortion caused by the errors. If important information contained in the packets, such as motion vectors, is not corrupted, the performance of the concealment technique can be considerably improved.

Recent video standards include a number of techniques to enhance the robustness of the compressed data streams [14]. Some techniques, such as the Reversible Variable Length Codes (RVLC), aim to extract the maximum amount of information from blocks of corrupted data. Others, for instance the insertion of resynchronization markers, help to recover the synchronization as fast as possible in case of errors. Data Partitioning (DP) and layered coding provide an a priori classification and separation of bits according to their sensitivity to errors and losses. These coding modes are usually coupled with Unequal Error Protection (UEP) schemes to take advantage of the different error sensitivity of the various classes or layers.

In this paper we argue that, for 802.11 networks, a partial checksum approach at the *link layer* coupled with data partitioning can improve the performance of video transmissions. The fundamental idea is that no error detection needs to be performed on perceptually less relevant bits, which are forwarded to the application as they are. Application level error resilient techniques are exploited to recover the maximum amount of useful information from the corrupted data. Note, however, that the MAC level checksum cannot be completely disabled due to the high bit error rate of wireless communications, hence the necessity of a partial checksum mechanism.

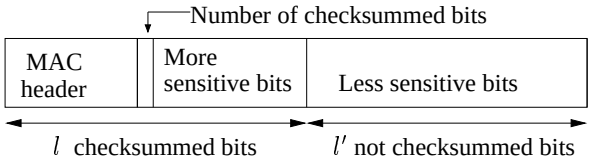

Fig. 1. Data-type 802.11 MAC frame structure modified for partial checksum.

MAC level headers, in fact, definitely need to be checked for errors to prevent misdelivered packets and to ensure the correct behavior of the 802.11 MAC protocol at each station. Moreover, higher level protocol headers, such as IP, UDP and RTP, need to be protected as well to ensure correct protocol operations.

We propose to design a flexible partial checksum mechanism introducing a two-byte fixed-length field before the payload that indicates how many bits from the beginning of the MAC frame must be considered to compute the frame FCS. The proposed frame structure is shown in Figure 1.

The UDP Lite protocol [5] considers a similar approach at the transport level. It introduces a partial checksum on the UDP data unit to protect only the UDP header and the sensitive part of the payload. If an error occurs in the checksummed bits, the receiver should drop the packet, otherwise it is forwarded to the application. The usefulness of the partial checksum, however, relies on the possibility to prevent the MAC layer from dropping corrupted packets at least in case of UDP Lite packets. Moreover, partial checksum at the transport level implies that packet checking is performed only at the transmission end points. Therefore, UDP Lite supports only end-to-end error control mechanisms, and it cannot provide fast hop-by-hop retransmission unlike the 802.11 MAC level partial checksum we propose. The benefit of fast hop-by-hop retransmission is particularly evident in scenarios characterized by a moderate or high end-to-end delay, in which end-to-end retransmission-based robustness techniques are inapplicable. In our proposal, UDP Lite is only used to allow a corrupted MAC payload to reach the application level.

IV. Partial Checksum Analysis

To illustrate the behavior of the proposed partial checksum technique, let us consider an 802.11 MAC frame where the checksummed and not checksummed amount of bits are respectively denoted by l and l' as in Fig. 1. For simplicity's sake, only for the analysis presented in this section, we assume that the bit error probability is uniform and equal to p, and each bit is independent of the others.

The packet error rate of a full checksummed packet (P^F_{err}) can, therefore, be expressed by the equation

$$P^F_{err} = 1 - (1-p)^{l+l'}, \tag{1}$$

where $l+l'$ is the total number of bits of which the packet is composed, including the MAC level header bits. If a maximum of $N-1$ retransmission are allowed, the Packet Loss Rate (PLR) of the full checksummed packets, P^F_{loss}, that is the probability of N unsuccessful transmissions, is

$$P^F_{loss}(N) = \left(P^F_{err}\right)^N = \left(1-(1-p)^{l+l'}\right)^N. \tag{2}$$

For a partial checksum scheme the decision to retransmit the packets is based only on the first l bits. Figure 2 shows a diagram of all the possibilities for a partial checksum scheme which allows up to N transmissions for each packet. In this scheme, the packet error rate (P^P_{err}) as measured at the MAC level is given by

$$P^P_{err} = 1-(1-p)^l. \tag{3}$$

Therefore, the PLR for the partially checksummed packets, P^P_{loss}, is given by

$$P^P_{loss}(N) = \left(P^P_{err}\right)^N = \left(1-(1-p)^l\right)^N. \tag{4}$$

This condition is indicated by the black circle marked as *lost* in Figure 2. Comparing Eq. (1) and (3), it is clear that, for a given p,

$$P^P_{err} < P^F_{err}. \tag{5}$$

Therefore, if the same number of retransmissions is allowed in both schemes, the packet loss probability of the partial checksum scheme is lower than the one of the full checksum scheme:

$$P^P_{loss}(N) < P^F_{loss}(N). \tag{6}$$

The Corrupted Packet Rate, P^P_{corr}, that is the probability that a packet is accepted with an error in the less sensitive l' bits for N transmissions (see the states labeled as *corrupted* in Figure 2), is

$$\begin{aligned} P^P_{corr}(N) = & \\ & (1-p)^l\left(1-(1-p)^{l'}\right) + \\ & \left(1-(1-p)^l\right)(1-p)^l\left(1-(1-p)^{l'}\right) + \\ & \cdots \\ & \left(1-(1-p)^l\right)^{N-1}(1-p)^l\left(1-(1-p)^{l'}\right) = \\ & (1-p)^l\left(1-(1-p)^{l'}\right)\sum_{i=0}^{N-1}\left(1-(1-p)^l\right)^i = \\ & (1-p)^l\left(1-(1-p)^{l'}\right)\frac{1-\left(1-(1-p)^l\right)^N}{1-(1-(1-p)^l)} = \\ & \left(1-(1-p)^{l'}\right)\left(1-\left(1-(1-p)^l\right)^N\right). \end{aligned} \tag{7}$$

For $N=1$, i.e. no retransmissions, the following equation holds

$$P^F_{loss}(1) = P^P_{loss}(1) + P^P_{corr}(1) \tag{8}$$

that is, some packets considered lost by the full checksum scheme are accepted by the partial checksum scheme because errors are located only in the not checksummed part.

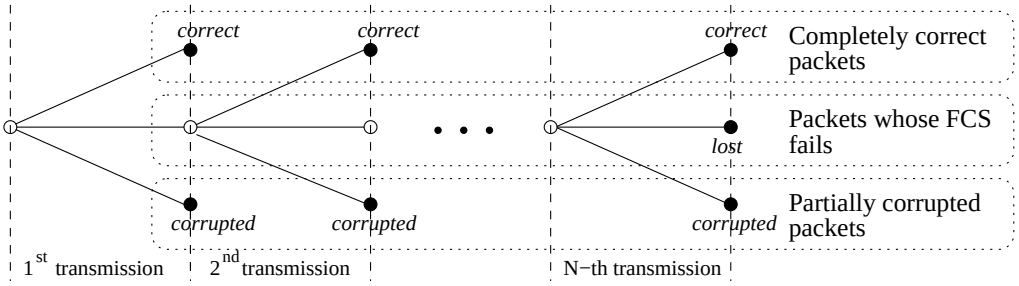

Fig. 2. Diagram of all the possible outcomes in case of N transmissions of a packet. Final states are shown as black circles.

Finally, consider the expected number of transmissions T for each packet, both in the full and partial checksum schemes. For the full checksum scheme, the expected number of transmissions with a maximum number of transmissions equals to N is given by

$$E\{T^{(F)}\} = \sum_{i=1}^{N-1} i \left(1 - P_{err}^{F}\right) \left(P_{err}^{F}\right)^{i-1} + N \left(P_{err}^{F}\right)^{N-1} = \\ = \frac{1 - \left(P_{err}^{F}\right)^{N}}{1 - P_{err}^{F}} = \sum_{i=0}^{N-1} \left(P_{err}^{F}\right)^{i}. \quad (9)$$

Analogously, for the partial checksum scheme, the expected number of transmissions is

$$E\{T^{(P)}\} = \sum_{i=1}^{N-1} i \left(1 - P_{err}^{P}\right) \left(P_{err}^{P}\right)^{i-1} + N \left(P_{err}^{P}\right)^{N-1} = \\ = \frac{1 - \left(P_{err}^{P}\right)^{N}}{1 - P_{err}^{P}} = \sum_{i=0}^{N-1} \left(P_{err}^{P}\right)^{i}. \quad (10)$$

Eq. (9) and (10) show that, for a given p, the number of packet transmissions is always lower in the partial checksum scheme. Each term of the sum, in fact, is less or equal than the corresponding term in the other sum (recall Eq. (5)). Therefore the network load due to the MAC level retransmissions is lower in the partial checksum case, with positive effects on the quality of service of all flows.

V. Partial Checksum for H.264 Video Transmission

In our experiments, the data partitioning functionality of the H.264 standard [15] is employed to pack the bits in sensitivity order. The compressed video stream is subdivided by the encoder into three classes or partitions. According to the standard, class A, the most important one, is used for headers, including macroblock headers and for motion vector information. Class B and C are designed to contain the texture information of the various types of macroblocks. In the proposed partial checksum scheme, all the data belonging to a slice is arranged into a single packet. Class A bits are placed at the beginning of the packet, and their number determines the checksum coverage. The remaining part of the packet is filled with class B and C bits. The proposed packet structure is shown in Figure 3(a).

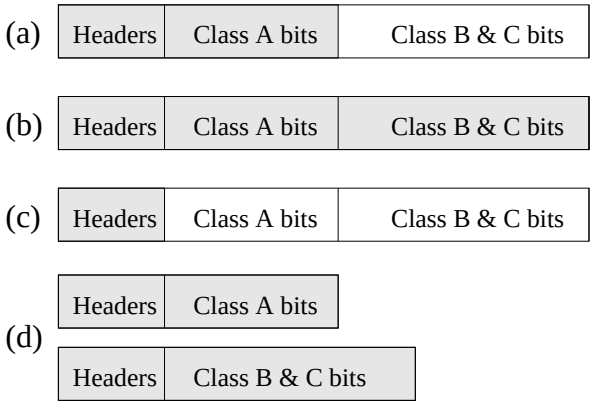

Fig. 3. Comparison of transmission schemes: proposed partial checksum (a), full checksum (b), header partial checksum (c) and UEP (d). The shaded area represents the checksum coverage. Both MAC and IP/UDP/RTP protocol headers are always protected.

To evaluate the performance of the proposed *partial checksum* scheme, Section VII presents comparisons with three other transmission schemes. Two schemes, namely the *full checksum* and the *header partial checksum*, feature a different amount of checksummed bits per packet. In the full checksum scheme, shown in Figure 3(b), all the bits of each packet are included in the checksum computation, hence that scheme represents the standard 802.11 behavior. The header-only partial checksum scheme, illustrated in Figure 3(c), is based on the same principle of the proposed partial checksum, but it restricts the checksum computation to the bits of the MAC and transport level headers. Finally, an *unequal error protection* scheme, shown in Figure 3(d), has been implemented for comparison purposes. It performs a checksum extended to the whole packets, but the H.264 partitions are placed into two separate MAC frames with different retry limits. Examples of such approach are described, for instance, in [3] [4]. To fairly compare the schemes, the same retry limit value as in our proposed scheme is assigned to packets containing the H.264 A-type partition, while packets containing the H.264 B- and C-type partitions are sent only once.

Note that only the full checksum scheme is fully compliant

with the current 802.11 standard; all the other schemes require either the partial checksum feature or the option to set a different retry limit for each packet class. Both features are not supported by the current 802.11 standard [1]. Proposals such as the 802.11e specification are expected to address some of these issues [16].

In all the schemes under test, we assume that any video unit (i.e. slice) whose class A bits are corrupted or lost cannot be decoded without introducing a large distortion, thus the whole video unit is considered lost. If the class A bits of a video unit are correctly received, the decoder faces two possible situations: the class B and C bits can be either corrupted or lost (depending on the transmission scheme). In the first case, the error resilience of the bitstream is exploited to decode the B and C partition as much as possible, while in the latter case a concealment technique can be applied taking advantage of the class A information, e.g. motion vectors.

VI. Simulation Setup

A. Encoder Configuration

Video sequences at QCIF resolution, 15 fps, are encoded with a fixed quantization parameters using the standard JM encoder, version 6.0a, modified to support standard-compliant data partitioning. The resulting bitrate is about 100 kbit/s. The PSNR value of the encoding distortion is 35.52 dB for the *foreman* sequence and 36.85 dB for the *carphone* sequence. Each frame is subdivided into three slices, each one corresponding to three consecutive rows of macroblocks. The mapping of the partitions into packets depends on the transmission schemes, as shown in Figure 3.

The following concealment techniques have been used. Packet losses are detected by means of the RTP sequence number. In case of loss of packets containing class A bits, the decoder applies a temporal concealment technique that replaces the missing pixels with the ones in the same position in the previous frame. Header partial checksum technique excepted, class A bits —if received— are always correct because they are checksummed. For the header partial checksum scheme, instead, class A bits could be corrupted. For this scheme, in our implementation the decoder discards the class A information if any syntax violation is detected during the class A decoding process.

When the class A bits of a certain video unit cannot be decoded, the decoder applies the temporal concealment technique and ignores any class B or C information for that unit. If the class A bits are correctly received and class B or C bits are available, the JM 6.0a decoder should take advantage of the error resilience techniques introduced by the encoder to improve the quality in case of errors. Nevertheless in our implementation, for simplicity's sake, we decided to discard the corrupted B and C bits and to simply take advantage of the motion vector information included in the class A bits to improve the concealment technique. In this case, the missing pixels are replaced with the ones in the previous frame as pointed to by the motion vectors. This method establishes the lower bounds of the performance of an error resilient decoder, because the not checksummed bits are not used at all to improve the quality of the received video, even when parts of them are correct. An improved error resilient decoder could, for instance, decode the not checksummed bits until it detects an error, and then discard the remaining part of the packet.

TABLE I

Parameters of the bit error trace.

Parameter	Value
Average burst length of errors (bit)	4.40
Error probability during a burst of errors	0.72
Average burst length of correct bits (bit)	16029
Bit error probability considering the whole trace	$1.97 \cdot 10^{-4}$

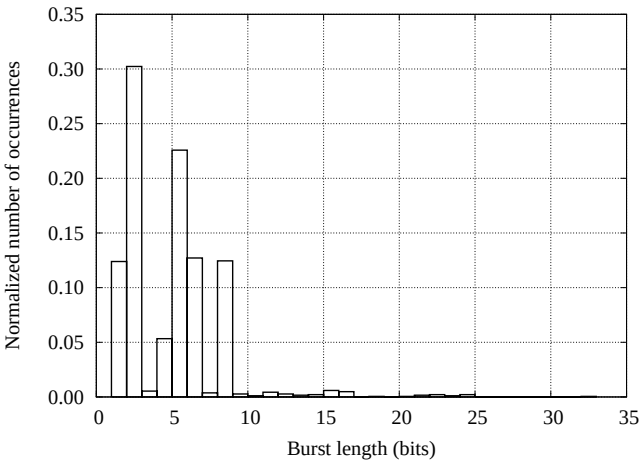

Fig. 4. Distribution of the burst error lengths computed over the trace used in the experiments.

B. Wireless Channel Modeling

Experiments have been conducted to obtain bit-level traces of an 802.11 wireless channel in different conditions. Packets with a known payload have been sent from a transmitter node to a receiver configured to operate in monitor mode. The receiver monitors the channel and it copies all the bits of all the received MAC frames to the user space regardless of the FCS indication. A bit error trace is then determined comparing the content of the received packets with the known payload. Traces capturing has been carried out in different conditions, such as varying transmitter-receiver distance, as well as presence of obstacles such as walls between them.

The traces present a clear bursty behavior when channel conditions are poor. Bursts are limited in length and include sporadically correct bits. In order to model the bursty behavior, we computed the distribution of burst length, both for bursts of errors and for burst of correct bits. The burst of errors has been defined as the sequence of bits, starting and ending with an error, followed by at least five correct bits.

Table I shows the parameters of the trace which served as basis for the channel model used in the simulations. The bit

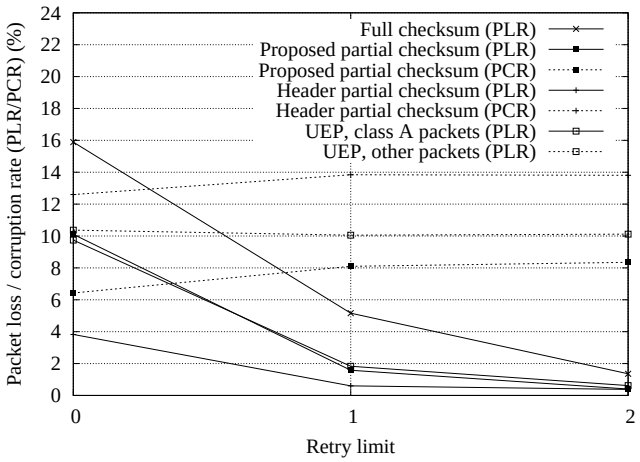

Fig. 5. Packet loss and corruption rate for the various transmission schemes. *Foreman* sequence.

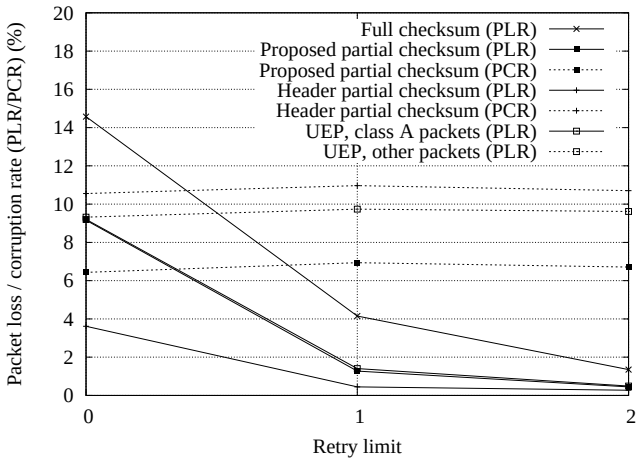

Fig. 6. Packet loss and corruption rate for the various transmission schemes. *Carphone* sequence.

error probability is computed as the number of erroneous bits over the total as well as the bit error probability limited to the bits belonging to a burst of errors as previously defined. Figure 4 illustrates the distribution of the error burst lengths. The other traces, representing different channel conditions, differ for the average burst length of correct bits.

Given the observed channel behavior, we decided to implement in the *ns* Network Simulator [17] a two-state model, in which each state represents either *good* or *bad* channel conditions. The model returns a value for each transmitted bit, according to its current state. No corruption is possible in the good state, while in the bad state bits are corrupted with the uniform probability shown in Table I (0.72). Each time the channel enters in one of the two states, a random variable determines the number of bits for which the channel will remain in that state. The probability density function of the two random variables —one for the bad and the other for the good state— is the one experimentally computed from the bit error traces.

C. Experimental Network Scenario

We considered an 802.11 wireless scenario in which a wireless node transmits a video sequence directly to a wireless receiver, in presence of other concurrent video traffic.

The *ns* network simulator [17] has been used to simulate the network behavior. For each radio channel, modeled as explained in Section VI-B, a different random realization has been used. The 802.11 network bandwidth is set to 11 Mb/s. Each video stream is sent using the IP/UDP/RTP protocol stack. The retry limit has been varied to evaluate its impact on performance of the proposed video transmission scheme. In the considered scenario, packets are dropped when queues are full or the MAC checksum fails. In the latter case, packets are retransmitted by the MAC layer until reaching the retry limit.

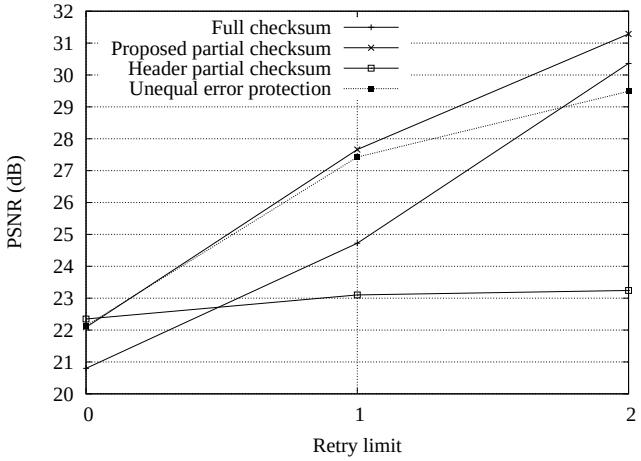

Fig. 7. PSNR performance of the various transmission schemes, for the *foreman* sequence.

VII. Results

This section presents a simulative analysis of the proposed partial checksum technique. Results are presented for the standard sequences *foreman* and *carphone*. The wireless channel parameters used in the simulations are shown in Table I.

In the considered scenario, the wireless transmitter sends packets directly to the wireless receiver without using intermediate hops. Figure 5 and 6 show, respectively for the *foreman* and *carphone* sequences, the packet loss and corruption rate as a function of the retry limit for different transmission schemes. Four other concurrent video flows that use the same transmission scheme are present in the network. Note that Equation (8) is experimentally verified. For the no retransmissions case, in fact, the amount of lost plus corrupted packets in the partial checksum scheme is equal to the amount of lost packets in the full checksum scheme. The same applies to the header partial checksum scheme. For higher retry limits,

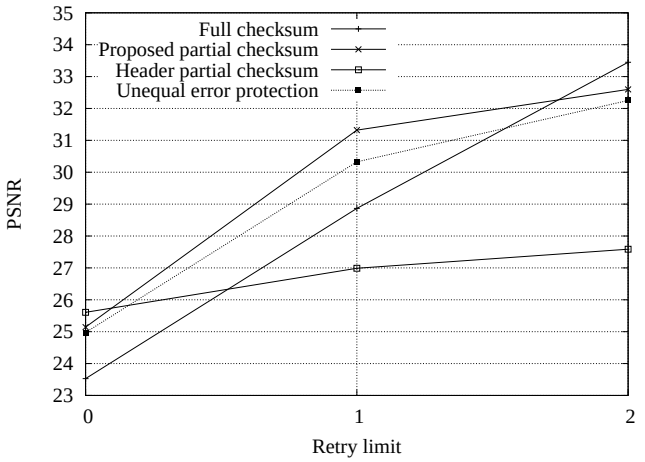

Fig. 8. PSNR performance of the various transmission schemes, for the *carphone* sequence.

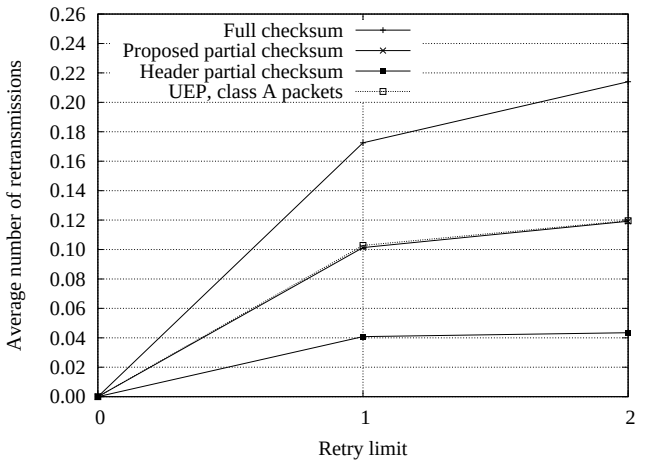

Fig. 9. Average number of retransmissions for the various transmission schemes. *Foreman* sequence.

the partial checksum scheme presents a significant packet loss rate reduction, while the number of corrupted packets slightly increases. Figure 5 also shows the packet loss rate of the unequal error protection scheme, separately for the two packet classes. Packets containing class A bits present a loss rate nearly equal to the partial checksum scheme, because the amount of protected bits is the same for both schemes. The packet loss rate of the other packets is approximately constant because they are transmitted only once.

The impact of the corrupted packets on video quality is now evaluated to assess the effectiveness of the proposed partial checksum scheme. Figure 7 and 8 compare, respectively for the *foreman* and *carphone* sequences, the PSNR performance of the four different transmission schemes as a function of the retry limit. Simulation conditions are the same as in Figure 5. The playout buffer is set to 150 ms. However, no packets are lost due to excessive delay. The average delay is similar for the various schemes except the unequal error protection, because this scheme transmits twice the number of packets. The higher average delay of the unequal error protection scheme originates from the higher number of channel contentions. Contention time is, in fact, not negligible in congested networks. The advantage of the proposed partial checksum scheme is clear for small values of the retry limit. In this condition, in fact, the negative effect on quality due to the high packet corruption rate is adequately counterbalanced by the strong loss rate reduction of the checksummed —hence important— packets. The performance of the full checksum scheme further confirms that the video decoder prefers corrupted packets to lost ones. However, if the retry limit value can be increased in order to recover most of the lost packets, the full checksum scheme becomes the preferred technique. Note however that the possibility to use a high retry limit implies a low network load, therefore plain transmission schemes are enough to provide good quality. The header partial checksum scheme consistently delivers a lower performance with respect to all the other schemes, hence confirming that partition A bits definitely need protection. The only exception is the zero retransmission case, in which, as expected, it is better to forward every received bit, even if corrupt, to the application level which decides about discarding.

Note that the presented PSNR values for the partial checksum scheme represent a lower bound of the performance achievable by the proposed scheme, since in our implementation the decoder does not exploit, for simplicity's sake, the correctly received parts of the B and C partitions contained in corrupted packets.

Finally, Figure 9 shows the average number of retransmissions per packet for the various transmission schemes. The full checksum scheme nearly needs twice the number of retransmission compared to the proposed partial checksum scheme. The unequal error protection scheme presents the same average retransmission number of the partial checksum scheme when considering class A packets only. Note, however, that the unequal error protection scheme needs to transmit twice the number of packets of the other schemes, so the actual network load is higher, due to the higher number of contentions. The header partial checksum scheme presents the lowest average number of retransmissions, but the performance of this scheme is less interesting due to the poor video quality it provides.

VIII. Conclusions

A partial checksum technique to enhance real-time video transmissions over 802.11 networks has been presented. A modification of the standard is proposed to limit the coverage of the link-layer checksum to the most important bits in the payload. A test implementation that exploits the data partitioning feature of the H.264 video coding standard has been described. In this scheme, video packets are retransmitted by the MAC layer only if errors are detected in protocol headers or in the error-sensitive bit class of the compressed video stream. The negative effects of errors on the less sensitive bits have been evaluated, showing that the lower number of

discarded video packets, lower average delay and reduced network load adequately counterbalance those errors. Network simulations performed using *ns* showed that the proposed partial checksum technique achieves consistent video quality improvements with respect to state-of-the-art techniques, such as unequal error protection based on different retry limits.

ACKNOWLEDGMENT

The authors would like to thank Antonio Servetti for his help to obtain the bit-level error traces and for the valuable discussions.

REFERENCES

[1] "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," *ISO/IEC 8802-11, ANSI/IEEE Std 802.11*, 1999.

[2] Q. Li and M. Van Der Schaar, "Providing adaptive QoS to layered video over wireless local area networks through real-time retry limit adaptation," *IEEE Transactions on Multimedia*, vol. 6, no. 2, pp. 278–290, April 2004.

[3] S. Krishnamachari, M. Van Der Schaar, S. Choi, and X. Xu, "Video streaming over wireless LANs: A cross-layer approach," in *Proc. Packet Video Workshop*, Nantes, France, April 2003.

[4] Y. Chen, J. C. Ye, C. R. Florjach, and K. Challopali, "Robust video streaming over wireless LAN with efficient scalable coding and prioritized adaptive transmission," in *Proc. IEEE Int. Conf. on Image Processing*, Barcelona, Spain, September 2003, vol. 3, pp. 285–288.

[5] L.-A. Larzon, M. Degermark, S. Pink, L.-E. Jonsson, and G. Fairhurst, "The UDP-lite protocol," *draft-ietf-tsvwg-udp-lite-02.txt*, August 2003.

[6] A. Singh, A. Konrad, and A. D. Joseph, "Performance evaluation of UDP lite for cellular video," in *Proc. Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Port Jefferson, NY, June 2001.

[7] S. A. Khayam, S. Karande, H. Radha, and D. Loguinov, "Performance analysis and modeling of errors and losses over 802.11b LANs for high-bit-rate real-time multimedia," *Signal Processing: Image Communication*, vol. 18, no. 7, pp. 575–595, August 2003.

[8] H. Zheng and J. Boyce, "An improved UDP protocol for video transmission over internet-to-wireless networks," *IEEE Trans. on Multimedia*, vol. 3, no. 3, pp. 356–365, September 2001.

[9] F. Yang, Q. Zhang, W. Zhu, and Y.-Q. Zhang, "An efficient transport scheme for multimedia over wireless internet," in *Proc. Int. Conf. Third Generation Wireless and Beyond*, San Francisco, CA, May 2001.

[10] G. Ding, H. Ghafoor, and B. Bhargava, "Error resilient video transmission over wireless networks," in *Proc. IEEE Workshop on Software Technologies for Future Embedded Systems*, Hakodate, Hokkaido, Japan, May 2003, pp. 31–34.

[11] A. Servetti and J. C. De Martin, "Link-level unequal error detection for speech transmission over 802.11 wireless networks," in *Special Workshop in Maui (SWIM), Lectures by Masters in Speech Processing*, Maui, HI, January 2004.

[12] H. Dong, I. D. Chakares, A. Gersho, E. Belding Royer, and J. D. Gibson, "Selective bit error checking at the MAC layer for voice over mobile ad hoc networks with IEEE 802.11," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, Atlanta, Georgia, March 2004.

[13] B.P. Crow, I. Widjaja, J.G. Kim, and P.T. Sakai, "IEEE 802.11 wireless local area netoworks," *IEEE Communications Magazine*, vol. 35, no. 9, pp. 116–126, Sept. 1997.

[14] Y. Wang and Q. Zhu, "Error control and concealment for video communication: a review," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 974–997, May 1998.

[15] ITU-T Rec. H.264 & ISO/IEC 14496-10 AVC, "Advanced video coding for generic audiovisual services," *ITU–T*, May 2003.

[16] IEEE 802 Committee, "Draft supplement to standard for telecommunications and information exchange between systems - LAN/MAN specific requirements - part 11: Wireless medium access control (MAC) and physical layer (PHY) specifications: Medium access control (MAC) enhancements for quality of service (QoS)," *IEEE Std 802.11e Draft*, July 2003.

[17] UCB/LBNL/VINT, "Network Simulator – ns – version 2," *URL: http://www.isi.edu/nsnam/ns*, 1997.