Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

An adaptive FEC to protect RoHC and UDP-Lite H.264 video critical data*

CHEN Zhi-kui

(*Networks and System Communications, Regional Computer Center, University of Stuttgart, Stuttgart 70550, Germany*)

E-mail: zchen@rus.uni-stuttgart.de

Received Dec. 10, 2005; revision accepted Feb. 20, 2006

Abstract: This paper describes how to use an Adaptive Forward Error Correction (AFEC) algorithm to efficiently protect the critical data areas of compressed headers and UDP-Lite packets for data transport in a radio link layer of a wireless connection. Augmented with RoHC and UDP-Lite, for H.264 video transmissions over wireless channels in a heterogeneous wired-wireless environment, the erroneous packet payloads can be useful and better able to cope with lost packets (native UDP case), by adopting some of the erasure and error resilient modes in H.264. The context transfer during inter/intra handover is also discussed. Simulations demonstrated that the proposed scenario significantly improves the PSNR performance and video quality.

Key words: Adaptive Forward Error Correction (AFEC), H.264, RoHC, UDP-Lite, UMTS, Bluetooth, Critical data, Erroneous SDU

doi:10.1631/jzus.2006.A0910 **Document code:** A **CLC number:** TN919.8

INTRODUCTION

Recommendation H.264 was approved by ITU-T in May 2003 and the approved draft specification is available for public review (Wiegand, 2003). The H.264 video codec has a very broad application range that covers all forms of digital compressed video from low bit rate Internet streaming applications to HDTV broadcast and digital cinema applications with near lossless coding. About 50% bandwidth would be saved by using H.264 when compared to the existing most efficient compression technology such as H263++ and MPEG-4 ASP. However, due to the likely business models in emerging wireless systems (3G/B3G) in which the end-user's costs are proportional to the transmitted data volume and also due to limited resources bandwidth and transmission power, compression efficiency is the main target for wireless video and multimedia applications. This makes H.264/AVC coding an attractive candidate for all wireless applications including Multimedia Messaging Services (MMS), Packet-switched Streaming Services (PSS) and conversational applications.

The most important characteristic of cellular links is the lossy behaviour, where a Bit Error Rate (BER) as high as 1e-3 must be accepted to keep the radio resources efficiently utilized. In severe operating situations, the BER can be as high as 1e-2. An additional problem is that the residual BER is nontrivial, i.e., lower layers can sometimes deliver frames containing undetected errors. For reducing bit errors and bandwidth, IEFT developed a robust RTP/UDP(-Lite) (Larzon and Degermark, 2004)/IP header compression scheme (Bormann *et al*., 2001). A viable header compression scheme for cellular links must be able to handle loss on the link between the compression and decompression point as well as loss before the compression point.

IETF RFC3095 (Bormann *et al*., 2001) provides three compression modes, U-mode (Unidirectional mode), O-mode (bidirectionnel Optimized mode) and R-mode (bidirectional Reliable mode). In U-mode, packets are sent in one direction only. Periodic re-

* Project (Nos. FP6-IST-2003-506997 and FP6-IST-2005-026943) supported partially by the European Information Society Technologies Framework Programme 6 Integrated Project: "Daidalos" (European IST Project: Daidalos, 2003~2008)

freshes are used to protect compressed headers against bit errors. O-mode uses a feedback channel to send error recovery requests and (optionally) acknowledgments of significant context updates. It reduces the number of damaged headers delivered to the upper layers due to residual errors or context invalidation. R-mode more intensively uses the feedback channel and a stricter logic at both compressor and decompressor to prevent loss of context synchronization between compression and decompression.

Bit errors cause individual headers to be lost or damaged. Some header impairments can cause context invalidation. Damage propagation and undetected residual errors both contribute to the number of damaged headers delivered to upper layers. In a noise and error prone wireless channel, assuming a BER of 1e-3 and with a compressed RTP/UDP-Lite/IP header size of 8 bytes, this would mean that for every 100 packets at least 6 would be damaged. U-mode has to refresh compressed headers using uncompressed headers with at least a 6% frequency at the packet level, so that header compression efficiency will decrease to at least 5%. On the other hand, due to randomicity of bit errors, periodic refreshes cannot prevent error propagation and loss propagation caused by damaged compressed headers. Also, the damaged packet has either to be delivered to the upper layer or discarded at the current layer. For O-mode and R-mode, they have to send at least 6 feedback packets to the compressor. O-mode can prevent error propagation and loss propagation, but the impaired packet is either delivered to the upper layer or discarded at the current layer. R-mode more frequently uses a feedback channel. High feedback rate will consume more network time and cause large delays. This will generate more packet losses. Of the other aspects, the frequent usage of the feedback channel will consume precious radio resources. This infringes on the aims of header compression or compression efficiency.

Therefore, by maximizing header compression efficiency we can expect to reduce refresh packets in U-mode and feedback packets in O and R modes. Using less redundancy to protect the compressed header is possible and can significantly improve end-to-end multimedia quality.

Meanwhile, the bit errors in the critical data coverage of UDP-Lite will cause checksum failures and damaged packet discards. For higher BER such as 1e-3 and 1e-2, bit errors in critical data coverage will discard a large number of packets. The error resilience probability provided by the H.264 compression standard provides maximum benefit and critical coverage data is also protected against bit errors.

Additionally, in the cases of multicast and Digital Video Broadcast (DVB), even including unicast, retransmission of the corrupted Service Data Units (SDUs) in L2 and the use of frequent feedbacks are impossible during real-time communications.

Therefore, this paper investigates the following aspects:

(1) An adaptive FEC algorithm to protect the critical data of L2 SDUs;

(2) AFEC data encapsulation for L2 PDUs;

(3) RoHC and AFEC configurations in a real-time communication system;

(4) Erroneous SDU delivery across network layers;

(5) Context transfer during handoff/handover.

This paper is structured as follows. Section 2 describes a self-adaptive FEC algorithm. Section 3 contributes to AFEC data encapsulation. RoHC and AFEC configurations are described in Section 4. Section 5 describes the erroneous SDU delivery across network layers. Context transfer due to handover is introduced in Section 6. Two examples, using UMTS and Bluetooth, are described in Section 7. Two trial simulations for non-real-time and real-time communication environment are given in Section 8. The paper is concluded in Section 9.

ADAPTIVE FEC ALGORITHM—GENERALIZED REED-SOLOMON ALGORITHM

An adaptive error control scheme for UDP-Lite and RoHC critical data in the PDCP sublayer for B3G/4G is designed according to the bit-error rate from the RLC in the receiver. For the FEC algorithm, a generalized Reed-Solomon (GRS) algorithm is proposed in the paper, whose details are described in (Chen and Christ, 2005a). The three parameters: FEC encoding rate, original data length and redundancy data length, decide a GRS code. Generally, the first two parameters decide the third. In a self-adaptive FEC algorithm system, the FEC encoding rate is decided by the BER of the receiving side and the previously used encoding rate.

At the sender, the initial FEC encoding rate (an integer expressed as a percentage, with a maximum value less than 127%) is set to 10%. From the second packet, the FEC rate is the maximal value of the previous FEC rate and the RBER (the received BER of the receiver).

At the receiver, there are three BERs. The first is the sender encoded value, the second is the PBER (the maximal value of all SDUs of the current PDCP PDU, as measured at the physical layer), and the third is that computed from the FEC decoding process. Thus, the receiver gets the BER, in terms of RBER, as the maximal value of the above three. If the current RBER is less than the previous value, the current RBER will not be transferred to the sender. Otherwise, the RBER will be delivered to sender.

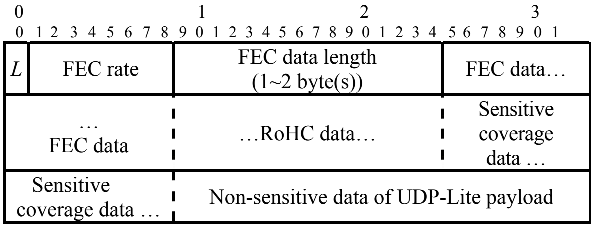
There are several ways to transfer the RBER from the receiver to the sender. First, RoHC can generate feedback to transfer the RBER to the sender. Second, a QoS managed system can deliver RBER to the sender (Chen and Christ, 2005b). In this case, the RBER is first delivered to a QoS client (normally the current mobile terminal), and then the RBER is delivered to the QoS broker (which manages and controls the communication quality) and then to the QoS manager (generally, the QoS manager is an access router, i.e., the sender in a wired-wireless communication system).

Note that AFEC is proposed to reduce error propagation and feedback channel usage in the RoHC. Also, to decrease packet loss, due to residual bit errors and limited retransmission in the physical layer, the maximum 127% FEC rate is enough.

PACKET FORMAT AND BASIC OPERATION

AFEC payload format

The payload format described here and shown in Fig.1, combines the original packet fields and the protected parity code produced by the compressed header, critical data coverage of UDP-Lite or both packet fields of an RTP/UDP-Lite (UDP)/IP packet. The format is described as follows: FEC encoding byte (1 bit for the redundancy length, 7 bits for FEC encoding bit rate), redundant data length (1 or 2 bytes) and redundant data are encapsulated at the front of the compressed header data.


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
L
FEC rate
FEC data length (1~2 byte(s))
FEC data…
… FEC data
…RoHC data…
Sensitive coverage data …
Sensitive coverage data …
Non-sensitive data of UDP-Lite payload

Where:

L bit: This bit indicates the parity code length. If *L* is 0, then the FEC redundancy length occupies one byte; if *L* is 1, then the FEC length occupies 2 bytes.

FEC rate: These 7 bits define the FEC encoding rate, the minimum value of 0 indicates no FEC; the maximum value of 127 represents the maximum FEC encoding rate of 1.27, i.e., if the original data is 100 bytes, then the encoded redundancy is maximal 127 bytes.

FEC length: 1 or 2 bytes to hold the encoded redundancy length.

FEC data: Redundancy data with FEC length bytes.

RoHC: The compressed packet header data and RoHC data.

Critical coverage data: The UDP-Lite checksummed data area.

Non-critical data of UDP-Lite: The UDP-Lite non-checksummed data area.

Fig.1 Protected RoHC packet header

At the server side, protection is applied before the packet is generated (i.e., before the Ethernet packet, PPP over Ethernet packet, or other packet generation over other wireless links such as BNEP in Bluetooth). The terminal will decode the received protected media packets after the Ethernet headers and/or PPP headers have been removed.

Critical data protection

The protection operation involves computing the parity (XOR) across the RoHC packet fields and UDP-Lite critical area which need to be protected. The recovery procedure allows terminals to correct the damaged RoHC and the critical data of UDP-Lite.

1. Critical data coverage protection of UDP-Lite

The recovery bit string is generated by the critical data of UDP-Lite. In H.264, the data in a packet is divided into two parts, a critical and a non-critical part; such as header and non-header parts. The critical parts to be checksummed can then be specified by the application in conformance with the data partitioning strategy, and the different levels of importance of the different segments within the packet according to their different error resiliencies; the usage of CABAC or CAVLC would lead to longer and shorter coverage respectively, e.g., in our tests, ‘partition A’ packets of the I and P or B frames in the three partitions of H.264 are checksummed whilst ‘partition B & C’ packets are partially checksummed. The decoding procedure at the terminal only recovers the required bytes inside critical coverage of UDP-Lite.

2. RoHC protection

The recovery bit string is generated by the RoHC header field. The terminal only recovers the damaged bytes inside RoHC header field.

3. Bit error protection of RoHC and critical data coverage of UDP-Lite

The recovery bit string is generated from both the RoHC header field and the critical data coverage of UDP-Lite. The decoding procedure at the terminal recovers the required bytes inside both the RoHC header field and the critical data coverage of UDP-Lite.

AFEC data encapsulation into an L2 PDU

AFEC data with payload format as shown in Fig.1, are encapsulated into a Radio Link Layer packet, which depends on the specific network technology. Two examples using UMTS and Bluetooth encapsulation are described in Section 7.

SETTING UP OF TRANSPORT PROTOCOL, AFEC AND ROHC

At session establishment, an offer/answer Session Description Protocol, SDP (Rosenberg and Schulzrinne, 2002), is used to describe user end-to-end configurations including application-specific quality of service of multimedia such as multimedia codec’s error resilient mode, transport protocol such as RTP, UDP, UDP-Lite, bandwidth/multimedia encode rate and RoHC/AFEC functionalities. Additionally, RoHC/AFEC configurations can also be implemented by the network’s initial connection, such as in Bluetooth (Chen and Christ, 2005a). Those distributed parts of the application may choose the mutually supportable configurations, which become actual configurations for the application.

ERRONEOUS SDU DELIVERY ACROSS NETWORK LAYERS

Delivery of erroneous SDUs determines whether error detection shall be used and if so, whether SDUs with an error in a sub-flow shall be delivered or not, i.e., to ignore the bit errors at Media Access Control (MAC) layer, which are not corrected by the physical layer due to residual bit errors and limited retransmission possibilities, the corrupted packets are relayed to the upper layer, namely the network layer.

For example, in 3GPP, whose QoS architecture provides some functionality to deliver erroneous SDUs (3GPP SP 23.107, 2005). There are three configurations: yes, no and ‘–’. If the configuration is set to ‘yes’, this implies that error detection is to be employed and that erroneous SDUs are to be delivered together with an error indication; if ‘no’, this implies that error detection is to be employed and that erroneous SDUs are to be discarded; and if ‘–’, this implies that SDUs are to be delivered without using error detection. For the case of variable protection, different sub-flows have different settings. When error detection configurations of a sub-flow are set to ‘no’, the erroneous SDU is discarded irrespective of the setting in other sub-flows. For an SDU with multiple sub-flows with the ‘yes’ setting, there may be one error indication per sub-flow, or, if there is only one error indication per SDU, it indicates that an error was detected in at least one of these sub-flows. More examples are described in Section 7.

In the network layer, there is no error checksum or erroneous PDU processing. The packet, whether correct or corrupted, will be relayed to transport layer.

The transport layer enables the UDP-Lite checksum, which is a partial data checksum. When bit errors are located inside critical-data areas, the packet will be dropped by UDP-Lite; otherwise, the packet will be transmitted to the application.

CONTEXT TRANSFER DURING HANDOVER

Context transfer is a technology to support the efficient handover and interoperable solutions for mobile services supported by Internet access network (Kempf, 2002) and to support integration of different wireless networks in Internet infrastructures based on interoperable services. In other words, the aim of context transfer is to efficiently re-establish the services in case of handovers without requiring the mobile host to explicitly perform from scratch all protocol flows for those services. The context transfer protocol (Loughney, 2005) is based on messages to initiate and authorize context transfer as well as messages transferring contexts prior to, during and after handovers.

Context transfer at a minimum can be used to replicate the configuration information needed to establish the respective protocols and services. In the draft, the RoHC and AFEC configurations are transferred to the new access router during handover.

On the other hand, context transfer also provides the capability to replicate state information, allowing state protocols and services at new nodes to be activated along the new path with less delay and less signalling overhead. In the case of RoHC and AFEC during handover, all RoHC and AFEC state information should be transferred to the new node, such as AFEC encoding rate, BER, RoHC compression profile, packet format, etc. and all previous header fields, which will be used to decode the coming RoHC header.

EXAMPLES

UMTS

1. Configuration setting up

An offer/answer Session Description Protocol, SDP (Rosenberg and Schulzrinne, 2002), is used to describe these user request specifications. The Session Initiation Protocol, SIP (Rosenberg *et al.*, 2002), is used to carry the SDP encapsulated user specifications to the called MT or the server.

The UDP-Lite usage information is configured using an offer/answer SDP to convey and exchange the configurations between caller and receiver of the call. Further details on an SDP description for H.264 are described in (Chen *et al.*, 2006). When the sender is requested to use UDP-Lite, a socket interface 'setsockopt' is activated to deliver the UDP-Lite protocol and multimedia UDP-Lite coverage to the kernel space. UDP-Lite coverage is required to be abstracted by the multimedia encoder.

2. Data encapsulation

The protected compressed data is encapsulated into PDCP as RoHC 3095 data. A PDCP PID (Packet Identifier) specifies the RoHC data which is protected (Table 1), where *n* is the number of PID values already assigned to other protocol packet types (3GPP TS 25.323, 2003).

Table 1 Mapping of PID values for RFC 3095 header compression protocol

PID	Optimization	Packet type
n+1	RFC3095	RFC3095 packet format
n+2	RFC3095	RFC3095 packet format with FEC

3. Erroneous SDU delivery

This could be configured by the application control. In the upper sublayer, the received erroneous SDUs will be reassembled into a PDCP PDU. Then, the corrupted packet will be delivered to the RoHC/AFEC sublayer; bit errors in the critical area of a received PDU will be corrected if possible. The recovered PDUs or PDUs without errors will recover its protocol overhead via RoHC decompression. If there is still a bit error in the decoded overhead, the decoded packet will be discarded and a feedback packet will be sent to the RoHC encoder to ask it to send a refresh frame and to increase the AFEC encode rate.

If the decode overhead is correct, the packet will be delivered to the transport layer via the IP layer as there is no error checksum in the IP layer.

The transport layer enables the UDP-Lite checksum, which is a partial data checksum. When bit errors are located inside critical data, the packets will be dropped by UDP-Lite; otherwise, the packet will be passed onto the application.

The application decoder will deal with the received erroneous packets using various error concealment techniques. Further, a context-based error detection and resilience approach, which deals with the bit error within erroneous packets, could be used to decode the erroneous packets; for more details see (Chen *et al.*, 2006; Daidalos Deliverable: D231, 2005).

Bluetooth

1. Configuration setting up

Similarly, RoHC/AFEC can be configured by SIP/SDP, if the initial establishment uses offer/answer SDP. On the other hand, it can also be configured by establishment of wireless connection. For example, in a Bluetooth initial connection, a Bluetooth user profile 'pand' tool can transfer the user

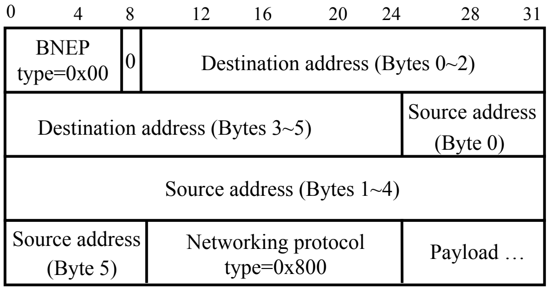
RoHC and AFEC configurations to the radio link layer at both MT and AR ends; for more details refer to (Chen and Christ, 2005a).

If the RoHC or AFEC configuration value is 0, this means that no RoHC or AFEC module is implemented. The RoHC configuration value corresponds to header compression profiles. AFEC configuration value corresponds to FEC encoding rate, with a maximum value of 127.

2. Data encapsulation

Bluetooth Network Encapsulation Protocol (BNEP) encapsulates packets from various networking protocols, which are then transported directly over the Bluetooth Logical Link Control and Adaptation Layer Protocol (L2CAP). BNEP removes and replaces the Ethernet Header with the BNEP Header. Finally, both the BNEP Header and the Ethernet Payload is encapsulated by L2CAP and is sent over the Bluetooth media (Bluetooth Core Specification 1.2, 2003).

A general BNEP Ethernet packet type header format is shown in Fig.2. Here the 8th bit of the first byte of BNEP packet describes whether it has an extension header. If its value is 0, that means there is no extension header, otherwise it has an extension header.



BNEP type=0x00 | 0 | Destination address (Bytes 0~2)

Destination address (Bytes 3~5) | Source address (Byte 0)

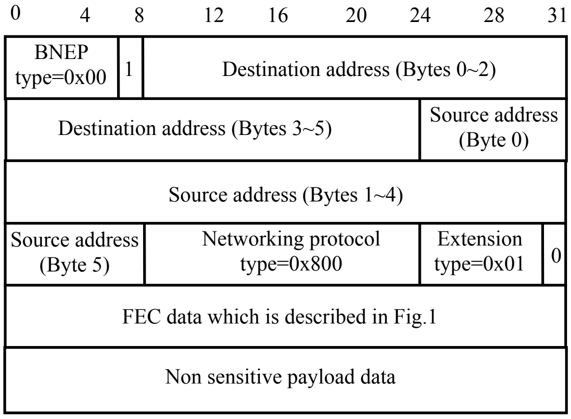
Source address (Bytes 1~4)

Source address (Byte 5) | Networking protocol type=0x800 | Payload …

Fig.2 BNEP general packet headers

AFEC payload encapsulation will use BNEP extension format. There are three extension types to encapsulate the protected data in BNEP packet format, which are:

(a) Only AFEC: BNEP has an extension for AFEC to protect uncompressed protocol headers and critical data, such as RTP, UDP-Lite/UDP, IP headers and UDP-Lite critical data (Fig.3). Extension Type is defined by 0x01; the 8th bit of the extension type byte is to state whether which has an extension header.



BNEP type=0x00 | 1 | Destination address (Bytes 0~2)

Destination address (Bytes 3~5) | Source address (Byte 0)

Source address (Bytes 1~4)

Source address (Byte 5) | Networking protocol type=0x800 | Extension type=0x01 | 0

FEC data which is described in Fig.1

Non sensitive payload data

Fig.3 AFEC and RoHC data encapsulation with an extension type in BNEP

(b) Only RoHC: BNEP has an extension for RoHC to protect the compressed protocol headers (RoHC data), whose Extension Type is defined by 0x02. The encapsulated BNEP packet has similar form compared with the case of "only AFEC".

(c) Both RoHC and AFEC: BNEP has an extension for AFEC and RoHC to protect the compressed protocol headers (RoHC data), whose Extension Type is defined by 0x03. Similarly, the encapsulated packet has similar format as that in Fig.3.

3. Erroneous SDU delivery

With Bluetooth, there are two ways to deal with erroneous SDUs. One is in the Baseband, and the other is provided by L2CAP (Bluetooth Core Specification 1.2, 2003).

In the Baseband, there are seven types of packets that could be used to deliver data from the upper layer to its peer. They are DM1, DH1, DM3, DH3, DM5, DH5 and AUX1. Except for AUX1, all have 16 bit CRCs generated over the payload. Thus, AUX1 can be used to deliver data that does not require CRC protection. Although other packet types provide CRC protection, it is possible for errored packets to pass the CRC check and, due to a residual bit error, they are still delivered to the upper layers.

An entity, namely, flush timeout, defined for the ARQ retransmission time, can be configured to some value, for which the default is infinite to re-transmit until physical link loss occurs, just like a 'reliable channel'. It is possible to produce an incomplete PDU and deliver it to the IP layer.

L2CAP provides enhanced error detection and retransmission capability to reduce the probability of undetected errors being passed up to the application layer and to recover from the loss of portions of the user data when the Baseband layer performs a flush on the ACL-U. In other words, if the delivery of erroneous L2CAP PDUs is enabled, the upper layer could receive the corrupted or incomplete L2CAP PDUs, which is depicted in (Bluetooth Core Specification 1.2, 2003).

SIMULATIONS

Simulations are implemented over a Bluetooth link. UDP-Lite protocol and RoHC/AFEC modules have been incorporated into a Linux Mandrake 10.0 (kernel version 2.6.9).

To consider dynamic changes in the bit error rate in the wireless communication environment, the widely accepted Elliot-Gilbert channel model is used. The Elliot-Gilbert channel model is basically a two state discrete time Markov chain (Fig.4). One state of the chain represents the Good-State (G); the other one represents the Bad-State (B). The state sojourn times and the bit error probability of every state are dependent on the bit error rate provided by the link budget analysis. By using the Elliot-Gilbert model we get time phases with higher bit error and lower bit error probabilities, which represents the bursty nature of the bit errors sufficiently.

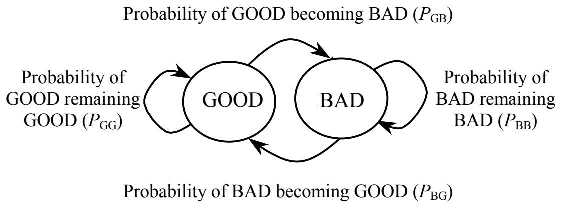

Probability of GOOD becoming BAD (P_{GB})
Probability of GOOD remaining GOOD (P_{GG})
GOOD
BAD
Probability of BAD remaining BAD (P_{BB})
Probability of BAD becoming GOOD (P_{BG})

Fig.4 Two-state Eliott-Gilbert channel model

The average bit error rate is then given by

$$P(\text{BER or bit error probability})=P_{GB}/(P_{BG}+P_{GB}).$$

Each slice of an encoded I and P or B frames has three partitions: partition *A* contains the headers of all coded MBs (Macro Block); partition *B* stores the intra Coded Block Patterns (CBP) and motion vectors and partition *C* contains inter coded block patterns and the coefficients. Two strategies of UDP-Lite checksum coverage have been considered for the encoded frame. In the first approach, the checksum is applied to the headers of partition *A*, *B* and *C*. In the second approach, the checksum covers entire partition *A* and the headers partition *B* and *C*. In the following figures, UDP-Lite1 and UDP-Lite2 stand for respectively the first and the second approach. Frames of type B cannot be used in the real time communications.

The RTP/UDP-Lite/IPv6 RoHC profile has been considered for header compression. The 60-byte header can be compressed into 8 bytes, i.e., 1 byte for RoHC header, 0 byte for compressed IP header (since packet size is less than 300 bytes), 4 bytes for compressed UDP-Lite header (2 bytes for UDP-Lite coverage and 2 bytes for UDP-Lite checksum), and 3 bytes for compressed RTP header. For convenience when comparing, UDP has the same RoHC header as UDP-Lite.

All simulations were carried out based on H.264 reference software. We would like to point out that the non-checksummed part of UDP-Lite partitioned packets are constructed CBP format (Wiegand, 2003): macroblock header+Delta Quantization+Coefficient of block 1, macroblock header+Delta Quantization+ Coefficient of block 2, …, macroblock header+Delta Quantization+Coefficient of block *n*, i.e., the macroblock header field is not checksummed by UDP-Lite. Bit errors may affect macroblock header and quantization coefficient, and the incorrectly decoded macroblock header will cause the decoder to make wrong decision on which block is to be decoded, therefore some modifications of the decoder should be made that ensures the decoder can successfully decode the received erroneous packets. The better way is to put slice header and all coded MB headers and quantization coefficients together in the packet header. That ensures all header information (slice header, MB header and quantization information, i.e., bit-error sensitive data) are checksummed. This paper uses the former scheme to deal with the erroneous packet.

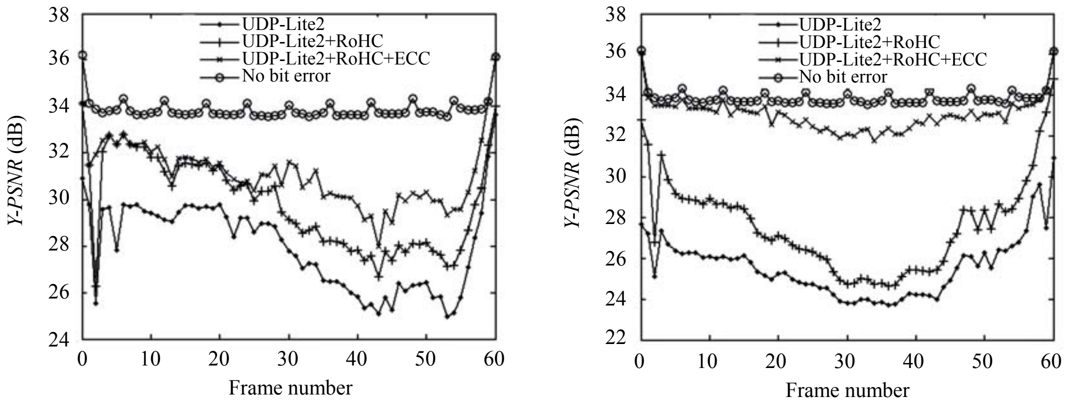
Non-real-time simulation

The Coastguard CIF sequence is used in the trials. The following H.264 error resilience modes have been used: slice mode with slice size of 200 bytes, three partitions per slice. Periodic I-frames refreshment has also been used. Each encoded group of pictures is composed of 60 frames with the following structure IBBBBBPBBBBB…PBBBBBB, i.e. being composed of one I-frame, 9 P-frames and 50 P-frames. Five reference frames have been used for prediction. The encoding rate is 30 frames per second (fps). Some PNSR performance and decoded video frames of UDP-Lite2 scheme are shown in Figs.5 and 6. In Fig.5, ECC means Error Correction Code, in which FEC encoding was initially setup to 10%. In our case, the redundant data of each protected packet is 2 bytes. In other words, improved PSNR performance and video quality is benefited from this two bytes redundancy. More simulations results can be found in (Chen and Christ, 2005a; Daidalos Deliverable: D231, 2005).

Real-time simulation

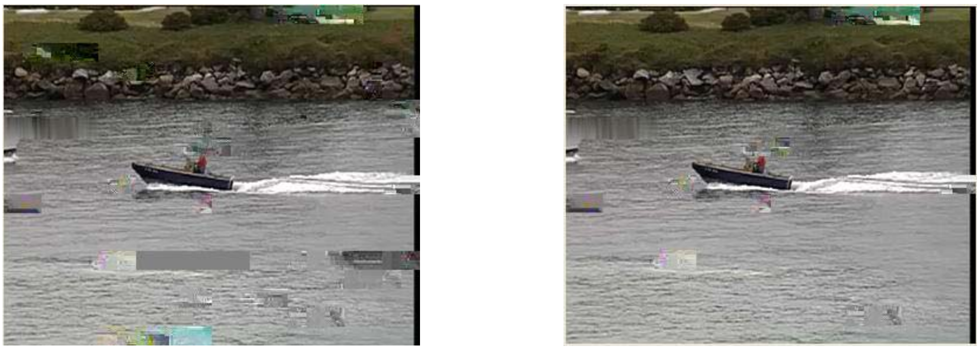
In the real-time communication environment, the B frame is not available. This experiment uses the coastguard CIF sequence as in the above experiment and each encoded group of pictures (GOP) is also composed of 60 frames with the following structure IP…PIP...P, i.e., being composed of one I-frame and 59 P-frames. Only one reference frame has been used for prediction. CAVLC entropy coding technique is implemented with an encoding rate of 15 fps. Only one UDP-Lite scheme is used, which is mentioned above as UDP-Lite2. Fig.7 shows the PSNR performance using different transport techniques at BER 5e-5.

Fig.8 shows the PSNR performance using different transport techniques at BER 1e-5.



(a) (b)

Fig.5 PSNR performance comparison of RoHC to RoHC plus FEC. (a) CABAC at BER 1e-4; (b) CAVLC at BER 1e-4



(a) (b)

Fig.6 Firstt frame of Coastguard sequence with CAVLC. (a) UDP-Lite2+RoHC at BER 1e-3; (b) UDP-Lite2+RoHC+FEC at BER 1e-3

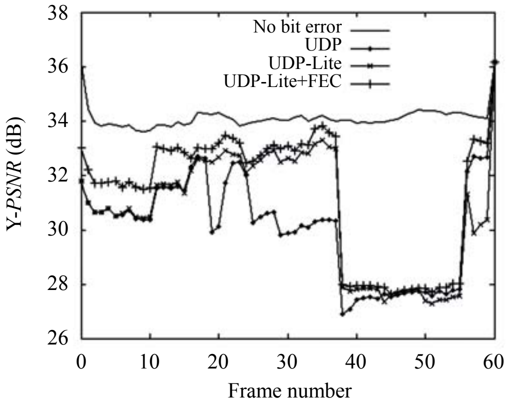


Fig.7 PSNR performance of first GOP with UDP-Lite and FEC at BER 5e-5

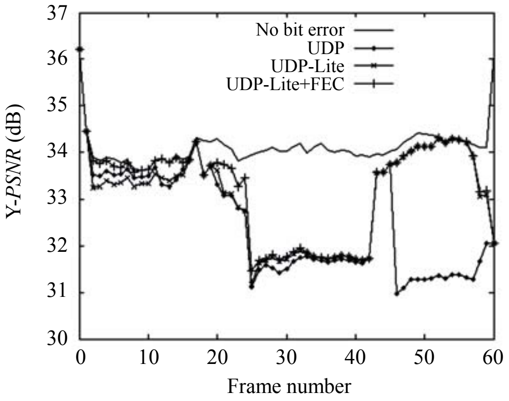


Fig.8 PSNR performance of first GOP with UDP-Lite and FEC at BER 1e-5

CONCLUSION

This paper proposes a self-adaptive forward error correction to improve end-to-end communications quality, which provides seamless QoS support over different wireless technologies for the access network. This includes cross-layer information exchange, robust header compression, packet payload partial checksum, erroneous SDU processing and delivery, context transfer during handover and QoS configuration. These techniques reduce packet losses due to residual errors and erroneous SDUs because of limited retransmissions at the physical layer, avoid error propagation in the RoHC decoding, reduce feedback channel usage in the RoHC, and to further decrease packet loss and packet delay.

The testing was applied to a Bluetooth communications scenario. The simulated results demonstrated that the PSNR performance and decoded video quality are significantly improved.

References

Bluetooth Core Specification 1.2, 2003.

Bormann, C., Burmeister, C., Degermark, M., *et al*., 2001. Robust Header Compression RoHC. IETF, RFC 3095.

Chen, Z., Christ, P., 2005a. Improving the Radio Link Layer QoS Performance for Bluetooth Real-time Video Communications. WTS2005. USA.

Chen, Z., Christ, P., 2005b. Improving Cross-layer QoS Performance in 4G Mobile Communications. GMC2005. China.

Chen, Z., Tang, Y., Christ, P., 2006. H.264 Based Video Transmission in Cross-layer Enable 4G Wireless Communication Systems. WWC2006. USA.

Daidalos Deliverable: D231, 2005. Deliverable on Simulations and Results. Http://www.ist-daidalos.org.

European IST Project: Daidalos, 2003~2008. Designing Advanced Interfaces for the Delivery and Administration of Location Independent Optimized Personal Services. (FP6-2002-IST-1-506997). Available at http://www.ist-daidalos.org.

Kempf, J., 2002. Problem Description: Reasons for Performing Context Transfers between Nodes in an IP Access Network. IETF, RFC 3374.

Larzon, L.A., Degermark, M., 2004. The Lightweight User Datagram Protocol (UDP-Lite). IETF, RFC 3828.

Loughney, J., 2005. Context Transfer Protocol (CXTP). IETF, RFC 4067.

Rosenberg, J., Schulzrinne, H., 2002. An Offer/Answer Model with the Session Description Protocol (SDP). IETF, RFC 3264.

Rosenberg, J., Schulzrinne, H., Camarillo, G., *et al*., 2002. SIP: Session Initiation Protocol. IETF, RFC 3261.

Wiegand, T., 2003. Final Committee Draft: Editor's Proposed Revisions. Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-F100.

3GPP TS 25.323, 2003. Packet Data Convergence Protocol (PDCP) Specification, (v6).

3GPP TS 23.107, 2005. Quality of Service (QoS) Concept and Architecture, (v6).