H.263 PICTURE HEADER RECOVERY IN H.324 VIDEOPHONE

Miska M. Hannuksela
Nokia Mobile Phones,
P.O.Box 68, 33721 Tampere, FINLAND
Tel: +358 40 5212845; Fax: +358 10 5057662
E-mail: miska.hannuksela@nokia.com

ABSTRACT

The scope of the H.324 videophone standard includes mobile networks which can in bad radio conditions be susceptible to higher bit error rates than most fixed networks. Different parts of the H.263 video bit-stream have unequal importance in video signal reconstruction. For example, the contents of a video picture cannot be decoded unless a so-called picture header has been correctly received or successfully recovered. In this paper we review methods for recovering corrupted H.263 picture headers. Moreover, we experimented how well two selected recovery methods operate in H.324 terminals connected to a simulated mobile network. The results showed that both tested algorithms play an important role in picture header recovery. In addition, we found that it is beneficial to set a certain parameter in the picture header differently than suggested in the H.263 recommendation.

1 INTRODUCTION

One of the goals in the design and standardization of mobile networks is to provide a sufficient quality of service for real-time audio-visual data transmission. The first services enabling mobile video transfer, such as GSM High Speed Circuit Switched Data (HSCSD), are already in use, and more advanced services are going to be launched during the next few years. Moreover, the standardization scope of video communication systems has included mobile networks and we can expect to see robust mobile video communication systems in the near future.

ITU-T recommendation H.324 specifies videophone services for low bit-rate circuit-switched networks. An H.324 terminal captures, compresses, and multiplexes the source signals into a single bit-stream and transmits this bit-stream over a network. While receiving a bit-stream, the terminal performs the opposite operations resulting into uncompressed output signals. Figure 1 shows the essential components of an H.324 terminal.

This paper is organized as follows: One of the key components in H.324 is the H.263 video coding recommendation introduced in section 2. Chapter 3 explains how transmission errors can be detected in H.324 terminals. After that, the paper concentrates on a specific area in H.263 video error resilience: picture header recovery. Section 4 reviews recovery algorithms, whereas section 5 presents simulation results. Finally, chapter 6 concludes the paper.

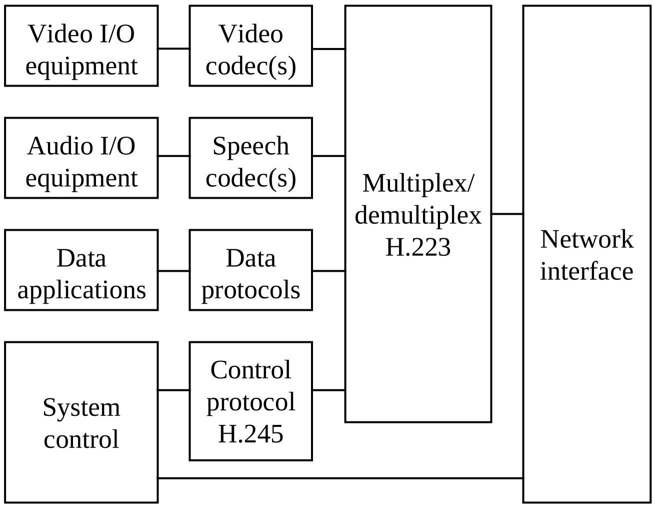

Figure 1. Components of an H.324 Terminal.

2 H.263 RECOMMENDATION

Currently, there are two finalized versions of H.263. Version 2 is an extension of version 1 providing several additional coding modes. H.263 version 3 will be formally adopted in November 2000, and it will contain three new optional annexes.

In H.263, each coded picture is divided into so-called macroblocks that are non-overlapping rectangles of 16 x 16 pixels. Moreover, a set of consecutive macroblocks in normal scanning order forms a picture segment. By default, a picture segment consists of an integral number of image-wide macroblock lines. If the optional slice structured mode is in use, a picture segment can contain any number of macroblocks.

Coded bit-stream can be roughly divided into picture segment headers and coded image content for picture segments. The first segment header of a picture is referred to as the picture header, and it contains parameters affecting decoding of the entire picture. Figure 2 shows the basic structure of H.263 bit-stream as well as simplified contents of the picture header and the picture segment header. The picture header contains among other things:

- A start-code which no other legal combination of codewords can emulate.
- Picture type, typically INTRA or INTER.
- Picture size.
- Flags telling which optional coding modes are in use.

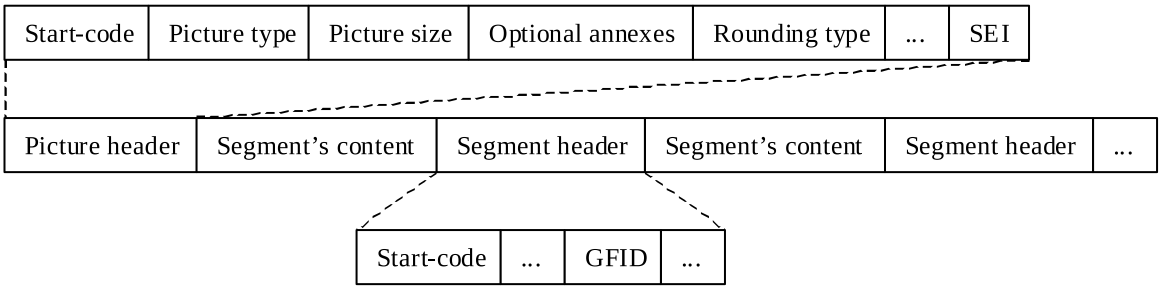

Figure 2. Simplified H.263 Bit-Stream Structure.

- Rounding type (RTYPE) for half-pixel-precision motion compensation. It was shown in [1] that it is beneficial to alter the rounding type in every other motion compensation reference picture in order to reduce rounding artifacts, and this recommendation is also included in the standard.
- Supplemental enhancement information (SEI) which does not affect the semantics for decoding a bit-stream. SEI enables enhanced features which can be harmlessly discarded by decoders that do not understand SEI codewords.

Each picture segment header begins with a unique start-code and contains a 2-bit Group of Blocks Frame Identifier (GFID). GFID has the same value within one picture. Moreover, GFID has the same value as in the previous picture, if certain vital fields in the picture header remain the same as in the previous picture. Otherwise, the value of GFID differs from the one in the previous picture.

3 DETECTION OF TRANSMISSION ERRORS

Error detection is an essential part of error control in a video communication system. In H.324, there are two levels to detect errors from video bit-streams: Firstly, H.223 demultiplexers have means to detect if a packet of video data contains bit errors. Secondly, H.263 decoders can detect bit errors, if they cause an illegal bit-stream syntax or semantics or if they cause unnatural behavior in the video signal.

The data flow between an H.263 codec and an H.223 (de)multiplexer is encapsulated in the so-called H.223 adaptation layer service data units (AL-SDUs). The multiplexer calculates a cyclic redundancy checksum (CRC) for each AL-SDU and includes it in the transmitted data. After transmission, the far-end demultiplexer recalculates the checksum, and if this checksum differs from the transmitted checksum, the data contains at least one error. Optionally, AL-SDUs may be associated with sequence numbers, which reveal if the demultiplexer is not able to recover all AL-SDUs.

The demultiplexer can either discard corrupted AL-SDUs or pass them to the decoder. In the former case, the decoder can restart decoding from the next start-code appearing in the video bit-stream. In the latter case, the decoder has to detect bit errors while processing the bit-stream. There are certain illegal bit patterns which cannot occur in a valid video bit-stream. In addition, there are semantic limitations, e.g., certain optional coding modes cannot be in use at the same time. Lastly, the decoder may utilize the characteristics of natural video signals to detect errors.

4 PICTURE HEADER RECOVERY

Video communication systems add redundancy to the transmitted data in order to enable recovery of corrupted or lost picture headers. In some systems, the transport coder can add such redundancy. For example, the RTP encapsulation of H.263 [2] allows repetition of vital picture header fields. However, H.223 does not provide means to protect picture headers, and therefore this paper concentrates on picture header recovery in source coding level.

This chapter contains the following sub-sections: Section 4.1 introduces GFID-based picture header recovery and discusses its weaknesses. Section 4.2 describes a complementary method based on picture header duplication.

4.1 GFID-Based Picture Header Recovery

If a decoder detects that a picture header is corrupted or missing, it can search for the next picture segment header. If the GFID of the found picture segment header is the same as in the previous picture, the picture data can be decoded using the picture header from the previous picture. Similarly, the decoder can wait for the next picture to arrive and check its GFID. If the GFID is the same as in the current picture, the decoder can use the picture header from the next picture to recover the lost or corrupted header.

In some cases, a picture header may be essentially different from the ones in the neighboring pictures, and therefore GFID-based picture header recovery does not work. These cases include:

- A change in picture type. Encoders may code an INTRA picture when the picture content changes rapidly or when users want to have a high-resolution snapshot picture.
- A change in half-pixel interpolation rounding (RTYPE).
- A change in picture size, which is typically caused when users request for a high-resolution snapshot picture.
- A change in which optional annexes are in use. The more optional coding modes used the more computational load in codecs. Thus, encoders may turn annexes on and off according to the available processing power at a given time.

If the lost picture header (and GFID) is different from the one in the neighboring pictures, the system can react in one or more of the following ways:

1. The decoder may request for an INTRA picture update from the far-end terminal using H.245 videoFastUpdatePicture command.
2. The decoder may guess which fields were changed in the picture header compared to the previous picture header. Then, it can try to decode the bit-stream. At the same time, the decoder should detect illegal syntax or semantics and unnatural video signal behavior. If the decoder finds that the bit-stream is invalid, it can deduce that it guessed the picture header incorrectly. This kind of an operation requires a sophisticated decoder implementation, and it is not always possible to guess the picture header right.
3. The terminals can use GFID numbering rules. For example, a picture type change from INTRA to INTER or vice versa may cause a GFID change as follows: $GFID_{new} = GFID_{old}$ XOR 01. Similarly, a rounding type change may result into $GFID_{new} = GFID_{old}$ XOR 10, and any other picture header change may cause an XOR operation by 11. If a picture header is lost, the decoder may be able to recover the picture header using the above mentioned rules. Anyway, these rules cannot represent all picture header changes, and since these rules are not listed in the standard, they can practically work only if the connected terminals are from the same manufacturer.

4.2 Repetition of Picture Headers in SEI

In order to avoid INTRA picture updates and decoding with wrong picture header parameters, we proposed an addition to the upcoming H.263 version 3 [3][4][5]. Recently, this addition was accepted as a part of draft annex W of H.263 version 3 [6].

When we designed our approach for picture header recovery, we considered the method included in MPEG-4 visual coding [7]. Each MPEG-4 picture segment header includes a 1-bit header extension code (HEC). If this bit is set, the picture segment contains a copy of the most essential fields of the picture header. This copy can be used to recover a lost or corrupted original picture header.

We wanted to design the picture header recovery feature in such a way that it is backward compatible with decoders following versions 1 and 2 of the H.263 recommendation. In other words, all decoders should be able to decode the bit-streams that include redundancy for picture header recovery. Thus, we could not introduce new codewords into picture segment headers, for example. Consequently, we could not adopt the MPEG-4 header extension as such.

Our method is based on a new function in H.263 supplemental enhancement information. The function enables repetition of the previous picture header in the supplemental enhancement information fields of the current picture. In case of a corrupted or lost picture header, the decoder operates as follows: At first, it carries out the GFID-based picture header recovery introduced in section 4.1. If the GFID-based recovery is not successful, the decoder looks at the next picture header. If the header contains a copy of the picture header, the decoder can replace the corrupted header with the copy.

The method inherits a considerable delay when recovering a corrupted picture header, since the recovery cannot take place until the beginning of the next picture is received. Since the operation of decoders is typically faster than real-time at least at low frame rates, decoders are likely to be able to catch up the time spent for waiting the next picture to arrive.

In order to overcome the delay problem, H.263 version 3 includes also a function to repeat the current picture header within the supplemental enhancement information. However, we assume that the usage of this feature is difficult due to the following reasons: Since typical bit errors are bursty, it is likely that the copy of the picture header is corrupted at the same time as the original picture header. If the original and the repeated picture headers differ, it is hard to deduce which one is correct. Furthermore, if the decoding synchronization is lost while decoding the original picture header, decoders cannot reliably resynchronize to the copy of the picture header since there is no start-code before the supplemental enhancement information.

5 SIMULATIONS

We used simulation conditions that are similar to the ITU-T common conditions for mobile H.324 simulation [8]. For convenience, we review the simulation procedure in the following paragraphs.

At first, we compressed a set of video sequences for a 64-kilobits-per-second (kbps) connection. We used a common test sequence called Foreman whose image size is 176 x 144 pixels. The source sequence (400 pictures) was looped in order to get long enough sequences comprising of 4000 compressed pictures at 7.5 pictures per second. Each loop began with an INTRA picture, and other pictures were INTER-coded. The previous picture header was repeated whenever it contained changes that disable GFID-based picture header recovery.

As we wanted to test picture header recovery characteristics, we used the following coding options that are crucial from the recovery point of view:

A. Rounding type (RTYPE) changes in each picture as suggested in the standard. No changes in annex usage.
B. No rounding type changes. Annex usage changes according to simulated fluctuations in processor load.
C. Constant rounding type. No changes in annex usage.

From the picture header recovery point of view, the number of GFID changes is essential: the less changes in GFID from picture to picture, the more probably GFID-based picture header recovery operates successfully. Table 1 shows how many times picture headers were repeated in the bit-streams. In other words, the table shows how many times the value of GFID was different from the ones in the neighboring pictures.

	A	B	C
Coded pictures	4000	4000	4000
Picture header copies	3999	704	40
Picture header copies (%)	100%	18%	1%

Table 1. Number of Repeated Picture Headers in the Example Bit-Streams.

Each compressed video file was processed by a program [9] that simulates an H.324-based system. The software encapsulates one picture segment in one H.223 AL-SDU and forms a multiplexed bit-stream. Then, the software corrupts the multiplexed bit-stream using a given bit error pattern that characterizes the network connection in use. We used two WCDMA error patterns corresponding to relatively bad radio conditions and having bit error rates of 9.73×10^{-5} and 9.73×10^{-4}. Finally, the software demultiplexes the corrupted bit-stream and discards erroneously received AL-SDUs. The resulting H.263 bit-streams were decoded, and the results are summarized in tables 2 and 3. Similar results were also obtained using other sequences than Foreman.

	A	B	C
Correctly received picture headers	3961	3959	3964
GFID-recovered picture headers	0	31	36
Picture header copy used	38	10	0
Unrecovered pictured headers	1	0	0

Table 2. Picture Header Decoding Statistics for 9.73×10^{-5} Bit Error Rate.

	A	B	C
Correctly received picture headers	3703	3707	3723
GFID-recovered picture headers	0	233	270
Picture header copy used	279	52	6
Unrecovered pictured headers	18	8	1

Table 3. Picture Header Decoding Statistics for 9.73×10^{-4} Bit Error Rate.

We can draw the following conclusions from the results:

- GFID-based picture header recovery is a powerful tool, if the value of GFID does not change frequently. Otherwise, picture header copies are essential.
- Changing the rounding type in each picture makes GFID-based picture header recovery ineffective. Thus, it results to more unrecoverable picture headers than when the rounding type is kept constant, as can be seen when comparing columns A and C of tables 2 and 3. Furthermore, the benefit from rounding type changes in bit-stream A was negligible due to two reasons: First, duplicate picture headers for each picture reserved a considerable amount of bandwidth. Second, rounding artifacts were not cumulated extensively due to frequent INTRA macroblock updating which is necessary in error-prone systems.
- Picture header repetition increases the size of the first picture segment. Thus, it increases the probability that the first segment contains bit errors and is discarded. This can be clearly seen in the number of correctly received picture headers in table 3 when comparing it to the number of repeated picture headers shown in table 1. However, when the bit error rate is moderate this behavior does not have a significant effect as can be seen from table 2.

6 CONCLUSION

Picture headers are important parts of H.263 video bit-stream. If a picture header is not correctly received or successfully recovered, the corresponding picture contents cannot be decoded. This paper looked at how to recover corrupted or lost picture headers in H.324 videophone terminals. Each coded picture is divided into segments, and each coded picture segment includes a Group of Blocks Frame Identifier (GFID) which tells if the current picture header is essentially similar to or different from the previous picture header. We showed that GFID-based picture header recovery operates well if there are no frequent changes in picture headers. Furthermore, we showed that picture header copies embedded into the H.263 bit-stream are useful in picture header recovery especially in bad channel conditions. The H.263 recommendation suggests that the half-pixel interpolation rounding type (RTYPE) should be inverted in each motion compensation reference picture. However, this is the most common cause for unsuccessful GFID-based picture header recovery, and we showed that it is beneficial to use a constant rounding type in error-prone environments.

REFERENCES

[1] Y. Nakaya, Y. Suzuki, and S. Misaka, “Avoidance of error accumulation in motion compensation”, ITU-T SG16 document Q15-A-35-R1, June 1997.

[2] IETF RFC2429, “RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)”, October 1998.

[3] M. Hannuksela, “Proposal for Error Resilience Function in Supplemental Enhancement Information”, ITU-T SG16 document Q15-H-12, July 1999.

[4] M. Hannuksela, “Core Experiment Description for Error Resilient Header Repetition”, ITU-T SG16 document Q15-I-25, October 1999.

[5] M. Hannuksela, M. Ledischke, and J. Zhang, "Results from Error Resilient Header Repetition Core Experiment", ITU-T SG16 document Q15-I-26-R0, October 1999.

[6] G.J. Sullivan (editor), "Determination Draft H.263++ Annexes", ITU-T SG16 document TD (P) 25, February 2000.

[7] ISO/IEC 14496-2, "Generic Coding of Audio-Visual Objects. Part 2: Visual", 1999.

[8] S. Wenger, “Common Conditions for Video Performance Evaluation in H.324/M error-prone systems”, ITU-T SG16 document Q15-G-46, July 1999.

[9] G.J. Sullivan, B. Wimmer, M. Luttrell, and S. Wenger, "Simple video packet mux simulator", ITU-T SG16 document Q15-G-42-R4, October 1999.