SMART SORTING OF H.264/AVC ENCODED SEQUENCES FOR APPLICATIONS OVER UMTS NETWORKS

Luca Superiori and Markus Rupp

Institute of Communications and Radio-Frequency Engineering
Vienna University of Technology
Gusshausstrasse 25/389, A-1040 Vienna, Austria
Email: {lsuper,mrupp}@nt.tuwien.ac.at

ABSTRACT

In this paper we propose a novel sorting mechanism suitable for H.264/AVC NALUs used in applications over UMTS networks. Each packet contains encoded information elements referred to a certain number of MBs. Following the standard, all the encoded information associated to a single MB is stored sequentially. In this work we propose to group the bits referring to a certain information element type even if associated to different MBs. The benefit of the proposed method will be discussed for different scenarios.

1. INTRODUCTION

The transmission of video sequences over the wireless networks is one of the most attractive improvements brought by the UMTS. The two most common applications are video telephony and video streaming. The Third Generation Partnership Project (3GPP) defines technical specifications for the Packet-switched streaming services. The H.264/AVC [1], the current state-of-art video codec, in its baseline profile is one of the video codecs recommended in [2]. H.264/AVC (Advanced Video Coding) was jointly standardized in 2003 by ITU-T VCEG (Video Coding Expert Group) and ISO/IEC MPEG (Moving Picture Expert Group) (as MPEG-4 Part 10).

The functionalities of the codec are conceptually subdivided in Video Coding Layer (VCL) and Network Abstraction Layer (NAL). The former deals with the proper video encoding operations, namely the translation of a video sequence (segmented into basic units called MacroBlocks (MBs)) into an encoded and compressed data stream. The latter ensures the stream to be *network friendly*, mapping the VCL output to the most appropriate transport or storage formats.

The applications considered in this work are dealing with the transmission of encoded video sequences over the IP (Internet Protocol)/RTP (Real Time Protocol)/UDP (User Datagram Protocol) protocols stack. The encoded video data contained in a packet defines a Network Abstraction Layer Unit (NALU). To each NALU the IP/RTP/UDP protocol headers are attached. The size of an IP packet is limited by the MTU (Maximum Transmission Unit) specified by the network. In order to limit the effect of lost or erroneously received packets, it is advisable to keep the size smaller than the MTU, considering the fact that the smaller the size of the packet, the higher the weight of the protocol headers [2].

In this work we describe a novel sorting mechanism for H.264/AVC inter-encoded frames. To improve the robustness of the transmission against channel errors and against network congestions, we propose to group the information elements (IE) depending on their type and not on the MB (MB) they are associated to. The proposed method ensures Data Partitioning (DP) and graceful degradation in case of incorrectly received video packets.

This paper is organized as follows. Section 2 offers an overview of the necessary concepts of video coding. Section 3 illustrates the proposed method. Section 4 and 5 describe the simulation setup and the results, respectively. The conclusions are drawn in Section 6

2. H.264/AVC INTER-ENCODED FRAME

Fixing the size of a packet, each NALU contains a variable integer number of encoded MBs, depending on the characteristics of the considered frame. Similarly to the other hybrid block-based video codec, two coding strategies are defined. Intra (I) encoded MBs exploit the spatial correlation between neighboring MBs belonging to the same frame. Inter (P) encoded MBs consider the temporal correlation between consecutive frames.

In practice, the inter prediction is far more efficient than the intra, resulting in inter encoded frames being 4 to 6 times smaller than intra encoded ones. On the other hand, inter decoded frames are highly sensitive to error propagation since they use the previously decoded frames as reference. Once a reference frame is incorrectly decoded, all the following

The authors thank mobilkom austria AG for technical and financial support of this work. The views expressed in this paper are those of the authors and do not necessarily reflect the views within mobilkom austria AG. Our thanks go to Wolfgang Karner for his contribution as well.

frames, even if correctly received, will contain visual impairments. The impairment propagation lasts until the next Intra encoded picture, as Intra encoded MBs contain only spatial references to MBs contained in the same NALU.

In a nutshell, the Inter encoding can be summarized as follows. For each MB, its best prediction is searched on the previously encoded pictures. In order to guarantee better prediction capabilities, the MB (16×16 pixels) can be subdivided in smaller blocks (up to 4×4 pixels). Once the temporal prediction has been performed, the DCT transformed and quantized difference (*residuals*) between the original and the predicted block is entropic encoded by means of CAVLC (Context Adaptive Variable Length Coding) when using the baseline profile.

The average size of the encoded information elements of Inter encoded pictures are depicted in Fig. 1. The considered

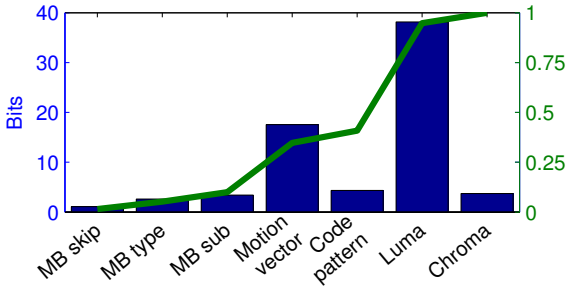

Fig. 1. Distribution of bits over IE types

IE groups [1] are:

- MB skip: Signalizes if a MB has been skipped or not.
- MB type: Defines which kind of subdivision has been performed (MB level).
- MB sub: Defines which kind of subdivision has been performed (sub-MB level).
- Motion vector: Indicates the relative position of the best prediction.
- Code pattern: Indicates which subMBs contain residual information.
- Luma: Represents the encoded luminance residuals.
- chroma: Represents the encoded chrominance residuals.

These elements are sequentially stored in the NALU MB by MB, as schematically shown in Figure 2. The interpretation of each encoded element depends strongly on the already decoded elements of different type associated to the same MB, as indicated by the solid arrows in Fig. 2. There is also a dependency between elements of the same type associated to different MBs, namely the motion vectors (MVs), as shown by the dashed arrows in Fig. 2. The encoded information about the MV of a block is the difference between its real

[1] Non relevant elements have been neglected for sake of simplicity

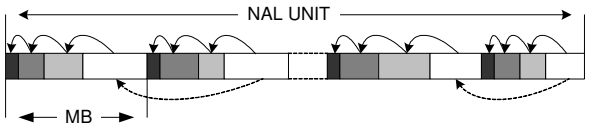

Fig. 2. Encoded elements within a NALU

value and the MV prediction. This prediction is calculated considering MVs associated to different MBs.

Therefore, we measure for encoded information element of an Inter frame a strong sequential intra-MB dependency (different elements belonging to the same MB) and a small inter-MB connection (elements referring to different MBs).

3. PROPOSED METHOD

In video transmissions over UMTS networks the common header stack consists of RTP over UDP over IP, forming a total of 40 Bytes of protocol overhead. The UDP contains a Cyclic Redundancy Check (CRC) indicating whether the received packet was affected by errors during the transmission or not. Usually, the UDP packets are discarded if the parity check fails [3]. This means that also a single erroneous bit can cause the discarding of the whole packet, even if it could have not caused any visible artifact in the decoded picture.

In [4, 5] the possibility of exploiting the correctly received part of a damaged packet was investigated. The results showed that the variable length entropic coding is highly sensitive to erroneous bits, causing *decoding desynchronization*. A single erroneous bit can cause the misinterpretation of the boundaries of all the following codewords up to the end of the packet. Therefore, all the MBs following the one where the error has been detected have to be concealed. The detection was performed noticing forbidden codewords or illegal action the decoder was driven into by desynchronized decoding.

This method was shown to be much more effective for Intra than for Inter encoded frames. An intrinsic difficulty of detecting syntax errors was measured in intra-predicted frames. Moreover, the effect of errors differs depending on the affected information elements. For instance, the `mb_skip_run` signalizes how many MBs, starting from the considered one, have to be skipped, i.e. can be replaced by the ones belonging to the previously decoded picture, considering the current motion compensation. While it is hard at the decoder side to detect whether the word is correct or not, the effects of errors in this element are relevant. Additionally to the possible decoding desynchronization, all the following encoded elements will be associated to erroneous MBs.

Due to this reason, we propose to store the various encoded elements contained in a packet differently. In our approach, all the information elements of the same type are stored sequentially, as shown in Figure 3. As discussed before, the standard encoding mechanism groups all the ele-

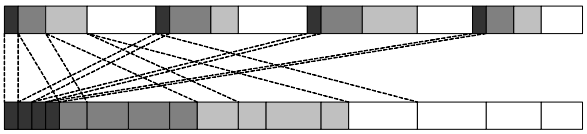

Fig. 3. Proposed sorting mechanism

ments referred to a single MB. The coding strategies used in the baseline profile allows the resorting of the encoded elements, grouping all the elements of the same type even if belonging to different MBs. Since all the dependencies between the elements are preserved, the method does not need any extra transcoding nor additional overhead. Possibly, an extra signalization bit in the slice header can indicate the use of the modified sorting mechanism. All the needed sorting operation can be applied to a standard compliant H.264/AVC encoded sequence. At the decoder side, the stream can be decoded as it is by a modified decoder or, without any additional transmission-rate overhead, unsorted first and then parsed by a standard H.264/AVC decoder.

The proposed mechanism allows the decoder to perform a progressive decoding of the packet. It begins with deciding whether the MBs stored in the packet have been skipped or not, then it considers the block and sub-block subdivision, it applies motion compensation and, at last, luminance and chrominance residual correction. The refining of the decoded picture slice is interrupted as soon as an error is detected, by means of strategies similar to the ones described in [4].

This approach can be considered as an *embedded* Data Partitioning (DP) within a single packet. As discussed in [6], one can differentiate the importance of the elements associated to the encoded MB. Namely, all the encoded elements up to the motion vector (partition A if using DP) allows the reconstruction of the picture. By means of luminance and chrominance residuals (partition C), such reconstruction is improved.

The bits indicating the basic characteristics of the encoded MBs are stored at the beginning of the packet. They are statistically more protected against decoding desynchronization. Considering the size of the packet being fixed to 700 Bytes and fixing the Bit Error Rate (BER) to 10^{-5}, the bars in Fig. 4 depicts the probability of all the elements of a certain type being correctly received.

Consistently with the distribution drawn in Fig. 1, the most important information are better protected than the residuals. Whilst carrying one of the most sensitive information, the `mb_skip_run` requires only the 1.52% of the code associated to a MB. Considering an average of 79 MB encoded in a packet, only 85 bits are necessary to signalize whether a MB has been skipped or not. Placing them at the beginning of the packet, this will ensure the greater statistical protection. In average, the "Partition A" elements require the 34.74% percent of the code. In other words, if the first 243 Bytes of the smart

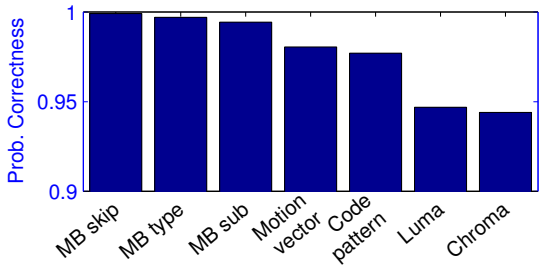

Fig. 4. Probability of IE groups being correctly received

sorted stream are correctly received, the whole picture can be reconstructed. Using the traditional encoding and sorting mechanism, this amount of bits would be sufficient to reconstruct (perfectly) 27 out of 79 MBs. The remaining 52 have to be concealed without additional informations.

4. SIMULATION SETUP

For simulation purposes, the whole method was implemented in Matlab. Using the H.264/AVC Joint Model (JM) [7] software as reference, the whole VCL functionalities of the decoder were reproduced as Matlab routines. The inputs of the script are the file in *264* format containing the encoded sequence and the trace file obtained after decoding[2] the *264* file. Despite all the necessary information is contained in the trace file, the *264* file has been used to simplify the reconstruction of the stream exploiting the already available elements that do not need any modification (SPS, PPS, encoded I frames and RTP header). A schematic representation of the implementation is shown in Fig. 5.

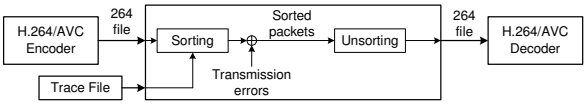

Fig. 5. Proposed simulation scheme

The trace file, together with the *264* file, are parsed by the sorting mechanism, returning a sequence of sorted packets. After transmitting over a noisy channel, each sorted packet is then unsorted. In order to unsort the packets, a whole H.264/AVC VCL decoder has been implemented in Matlab. The unsorted P packets, together with the SPS, PPS and I frames are reassociated to the respective RTP headers and given as input to a standard H.264/AVC encoder.

The error detection was implemented in the unsorting block. Once an error has been detected, all the following information elements are assumed to be invalid and the produced *264* file

[2]We were forced to use the *decoder* trace rather than the *encoder* (as one may expect) since the implementation of latter in the JM contains several software bugs.

does not contain any further element. For instance, when detecting an error in the residuals, the coded block pattern of all the following MBs is modified in a way that no residuals are associated to the considered MBs. In general, as soon as one error has been detected when decoding the IE type $\mathbf{f}_i$ of the MB MB_j, all the following MBs $\{\mathrm{MB}_k | k \geq j\}$ will not contain the IE type $\mathbf{f}_i$, and all the MBs of the packet will not contain the IE types $\{\mathbf{f}_k | k > i\}$.

5. RESULTS

The performance of the proposed method was compared with the one obtained using the approach described in [4]. The considered video sequence was the *Foreman* in QCIF resolution, encoded in baseline profile. The quantization parameter was set constant to 26.

The two compared methods share the same error detection mechanism. The comparison was performed analyzing the quality, measured in terms of luminance Peak Signal to Noise Ratio (Y-PSNR) (mappable to the Mean Opinion Score (MOS) as described in [8]), of the two approaches when detecting an error in the same bit position. Various test were performed over frames characterized by different spatial and temporal attributes. Figure 6 shows the results obtained decoding a frame of Foreman with moderate camera movement.

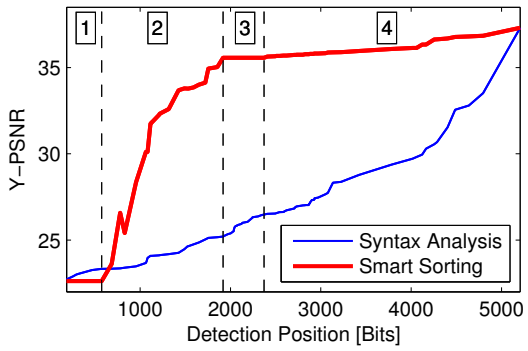

Fig. 6. Detailed comparison for frame 3

By using the method proposed in this article, the decoding of a packet can be subdivided in four different regions, labeled as in Fig. 6. The first one comprises the decoding of the IEs indicating the MB skip, MB type and MB sub. If an error is detected in this region, the whole frame is concealed, since there is no useful information for the reconstruction available. This corresponds to about 7 MBs perfectly decoded using the method described in [4]. In the second region are stored the MVs. As soon as an error is detected in this region, only the following MBs are concealed, whereas for the previous one the correctly received MVs are used. The third region encloses the `code block pattern` elements, indicating which subMBs contain residuals. This IE does not carry any intrinsic information itself, therefore in the region three we do not measure any quality improvement. Finally, in the region four are stored the luminance and chrominance residuals. Despite the size of the code associated to them (65% of the whole code), the picture refinement they allow is not that considerable as the one brought by the MVs.

Moreover, the plot associated to the method in [4] shows a non regular trend and sudden rises, as the one at the end of the packet. This is because the weight of the different MBs for the resulting quality is not constant. Exploiting the proposed smart sorting mechanism, the MVs of the MBs stored at the end of the packet are decoded sooner. Additionally, the effectiveness of the temporal concealment is increased, since the probability of having neighboring MBs with valid MVs is higher.

6. CONCLUSIONS

In this paper a novel sorting algorithm for H.264/AVC encoded sequences is presented. Without increasing the rate nor needing the transcoding of the sequence, the information elements are grouped by type rather than by macroblock. The most important groups are stored at the beginning of the packet, where, statistically, they are more protected against decoding desynchronization. This allows for a progressive packet decoding, increasing the probability of having valid information for the whole picture. The effectiveness of the method has been assessed using the standard sequential decoding as reference.

7. REFERENCES

[1] ITU-T Recommendation H.264 and ISO/IEC 11496-10 (MPEG-4), "AVC: Advanced Video Coding for Generic Audiovisual Services", version 3, 2005.

[2] 3GPP Technical Specification 26.234, "Transparent end-to-end Packet-switched Streaming Service (PSS), Protocols and codecs" (Release 6)

[3] S. Wenger, "H.264/AVC over IP," IEEE Trans. on Circuits and Systems for Video Techn., vol. 13, no. 7, pp. 645-656, Jul. 2003.

[4] L. Superiori, O. Nemethova, and M. Rupp, "Performance of a H.264/AVC error detection algorithm based on syntax analysis," in Proc. of Int. Conf. on Advances in Mobile Computing and Multimedia (MoMM) 2006, Indonesia, Dec. 2006.

[5] O. Nemethova , W. Karner, A. Al Moghrabi , M. Rupp, "Cross-Layer Error Detection for H.264 Video over UMTS," in Proc. of Int. Conf. on Wireless Personal Multimedia Communications (WPMC), Aalborg, Denmark, Sep. 2005.

[6] Stockhammer, T.; Bystrom, M., "H.264/AVC data partitioning for mobile video communication," in Proc. of Int. Conf.e on Image Processing (ICIP) 2004, vol.1, pp. 545-548,Oct. 2004

[7] H.264/AVC Software Coordination, "Joint Model Software," ver.10.2, available in `http://iphome.hhi.de/suehring/tml/`.

[8] O Nemethova, M Ries, E Siffel, M Rupp, "Quality Assessment for H. 264 Coded Low-Rate and low-Resolution Video Sequences" in Proc. of Conf. on Internet and Inf. Techn., 2004.