Video Quality Experts Group

21-24 January 2014
Boulder, CO
English only
Original: English

JEG-Hybrid Contribution

Author: Enrico Masala, Politecnico di Torino, Italy – Email: enrico.masala@polito.it

Date: Jan 16, 2014

Title: Simulation of Robust H.264/AVC Decoding in presence of Data Loss

Introduction

Video decoding software often crash when processing compressed bitstream data corrupted due to missing parts, especially if the amount of lost data is large or affect consecutive elements. Making the software robust to any loss pattern typically requires complex modifications to the source code. Also, software for which source code is publicly available is often not well written and poor documented, hence difficult to modify. This document proposes an approach to simulate the behavior of a robust H.264/AVC [1] decoder in presence of any data loss pattern, with the final aim to compute the degraded video signal resulting from the decoding operation.

Aim and context of the proposal

The free availability of an H.264/AVC decoder immune from crashes regardless of the data loss pattern would allow to enlarge the JEG-Hybrid database [2], currently composed of 12,960 compressed video streams with different coding artifacts, by applying data loss patterns similar to those encountered in transmissions over packet networks. This could significantly expand the type of artifacts considered in the database.

Rationale

In order to make the software robust to crashes and at the same time accurately reproduce the effects of the losses, a simulation of the concealment technique is performed during the decoding process of the original, uncorrupted compressed video sequence, in a way that does not affect the internal state of the decoder apart from the content of the decoded picture buffer (DPB).

Every time a video element (e.g., a NALU) that should be considered lost is encountered, the content of the DPB is modified on-the-fly to apply the concealment technique to the areas of the picture that should be affected by the losses. For instance, a "copy concealment technique" overwrites the content of the picture in the buffer with the content of the corresponding area of another picture which has already been decoded. After this modification, which is executed each time a new element is supposed to be lost, the decoder continues its normal operations not to disrupt its internal state that, if incorrectly managed, may lead to crashes due to unhandled cases. This

approach has already been used by the author in a number of research papers dealing with multimedia communications over packet networks [3][4].

Discussion

The main advantage of such an approach is:

- being able to avoid crashes due to data loss (any loss pattern is supported);
- different concealment techniques can be easily implemented;
- realistic simulation of the reconstructed video by a decoder operating on a real corrupted bitstream.

It should be noted that in most of the cases the reconstructed video is exactly the same as the one that would be produced by a decoder operating on a real corrupted bitstream using the same concealment technique. In very few cases there might be a slight misalignment, e.g., when some coding modes that require the availability of data from previous pictures incorrectly assume that the data is available when it would not be according to the loss pattern. An example could be the computation of motion vector predictors for Direct Mode in H.264/AVC that may be different in the two cases, leading to slightly different reconstructed videos.

In any case, the type of artifacts generated by this approach are very similar, if not equal, to the ones of the decoder operating on a real corrupted bitstream. The possibility to consider any loss pattern and realistically computing the corresponding artifacts in the decoded video is the main reason why this software is deemed to be useful for the JEG-Hybrid project.

Details about the software

The software is available as a modification of the reference decoder included in JM 16.1 test model software [5], implementing a "copy-concealment" technique applied on a NAL unit basis, using as a reference the decoded picture that immediately precedes the concealed one in decoding order (note that this may include pictures not marked as reference in the standard H.264/AVC decoding process). The software handles any compressed format supported by JM 16.1 (e.g, Annex B file format, or RTP file format used by the JM software).

The software has been tested successfully by the author of this document on the entire JEG-Hybrid dataset (12,960 sequences, generated by both the JM encoder and the x264 software [6]) without incurring into crashes while applying randomly generated NALU loss patterns with 20% independent loss probability.

The software will be made available, with source code, before Jan 22, 2014, at the URL: `http://media.polito.it/jeg`

Usage

The software retains all the features and characteristics of the original JM 16.1 software [3], in particular the original command line parameters and the format of the configuration file. However, the following parameters have been added to support the described operating mode:

`-d configfile` : to simplify parameter parsing, the decoder configuration file must be preceded by the `-d` parameter (differently from the original JM 16.1 software where no flag was necessary) when using any of the following parameters.

`-loss losspattern` : if the parameter is present, it takes as argument a text file which contains the loss pattern to be use used in the simulation, in the form of a list of events. Each event must be a '`0`' or '`1`' (ASCII characters representing zero or one), followed by a newline. Each line represent respectively no loss (0) or presence of loss (1). Every line in the `losspattern` is matched to a NALU in the compressed bitstream. The match happens in order, one to one: the first NALU in the bitstream corresponds to the first line of the file, etc. In case the display order and coding order differs, e.g., when B-type slices are present in the compressed bitstream, the coding order is used. In other words, the order of the NALU in the bitstream is used to match NALUs and events in the `losspattern`. If more events than NALUs are present in the `losspattern`, the remaining data in the `losspattern` are discarded. If less events are present, the decoder terminates with an error message.

`-firstIsLost` : parameter, without argument, that specifies if losses must be applied to the first picture or not, regardless of the values present in the `losspattern`. Default behavior is to not apply losses to the first picture, since no previous pictures are available. If the parameter is present, in case the `losspattern` indicates that a NALU in the first picture is lost, the corresponding area is concealed using a fixed value, equal to 128 for the luminance channel and 0 for chrominance channels (corresponding to a mid-grey color). Note that, even if the parameter is not specified, events are consumed from the `losspattern` for the purpose of matching NALUs and events. Therefore, the number of events consumed from the `losspattern` always corresponds to the number of NALUs in the compressed bitstream.

Usage Example

Assume that the content of `losspattern.txt` is:

```
1
1
0
```

and the considered compressed bitstream is composed by three pictures, one NALU per picture, with the following encoding pattern: I, B, P (display order), i.e., I, P, B (coding order).

Example 1:

```
ldecod.exe -d decoder.cfg -loss losspattern.txt
```

produces a reconstructed video sequence (in the file specified as usual in `decoder.cfg`) with a simulated loss of the whole P picture (and the corresponding error propagation effects on the B picture).

Example 2:

```
ldecod.exe -d decoder.cfg -loss losspattern.txt -firstIsLost
```

produces a reconstructed video sequence (in the file specified as usual in `decoder.cfg`) with a simulated loss of the whole initial I picture (concealed using the mid-grey value) and the P picture (and the corresponding error propagation effects on the B picture).

References

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

[2] JEG-Hybrid Database, available at: ftp://ftp.ivc.polytech.univ-nantes.fr/VQEG/JEG/HYBRID/database/

[3] E. Masala, A. Vesco, M. Baldi, J.C. De Martin, "Optimized H.264 Video Encoding and Packetization for Video Transmission Over Pipeline Forwarding Networks", IEEE Transactions on Multimedia, vol. 11, n. 5, Aug 2009, pp. 972-985

[4] E. Masala, F. De Vito, J.C. De Martin, "On the Effects of Sender-Receiver Concealment Mismatch on Multimedia Communication Optimization", accepted for publication in Multimedia Tools and Applications 2013, DOI: 10.1007/s11042-013-1751-y

[5] Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG. Joint Model Number 16.1 (JM 16.1) Available: http://iphome.hhi.de/suehring/tml/download/old_jm/jm16.1.zip

[6] VideoLan Organization, "x264 Home Page", http://www.videolan.org/developers/x264.html