

# H.26L over IP: The IP-Network Adaptation Layer

Stephan Wenger

[stewe@cs.tu-berlin.de](mailto:stewe@cs.tu-berlin.de)

**Abstract:** H.26L, the ITU-T's forthcoming new video compression standard, conceptually consists of a Video Coding Layer that performs the tasks traditionally associated with video coding, and a Network Adaptation Layer (NAL) that implements video-specific support features for a variety of networks. This paper describes the generic H.26L NAL, which includes revolutionary mechanisms such as the Parameter Set concept – the asynchronous transmission of header information out-of-band – or the packet structure of the H.26L coded video. Such concepts are described in some detail, and are augmented with a limited number of simulation results that verify the efficiency of some of the coding tools introduced in the NAL.

## 1. Introduction

Since 1997, the ITU-T's Video Coding Experts Group (VCEG) has been working on a new video coding standard with the internal denomination H.26L. While during the first few years not much progress could be reported, this project got momentum with the finalization of version 3 of H.263 (aka. as H.263++) in late 2000. Since no version 4 of H.263 is planned, the focus of the VCEG group has shifted from H.263 towards the H.26L project, and since then much progress can be reported.

In late 2001, MPEG's video group and VCEG decided to work together as a Joint Video Team (JVT), and to create common text for a forthcoming ITU-T Recommendation and for a new part of the MPEG-4 standard based on the then current working draft of H.26L [1]. The timeline of the joint project requires a technically frozen specification by the end of a meeting in May 2002. Hence, the design now quickly converges towards a working specification.

One of the main design goals of the H.26L project is to allow for a seamless and easy integration of H.26L coded video into all current protocol and multiplex architectures – a goal that was summarized as “Network Friendliness”. Undoubtedly, the Internet or, more generally, IP networks will carry a significant amount of H.26L video and, hence, being “network friendly” to IP-based networks is one of the main design goals of the H.26L project.

H.26L consists of two conceptually different layers.

- The Video Coding Layer (VCL) contains the specification of the compression engine – mechanisms such as motion compensation, transform coding of coefficients, and entropy coding. The VCL is in so far transport unaware, as the created bit strings – note that the word bit stream is not appropriate, which should become clearer later – are transported unchanged over the network, and even across network boundaries. The highest data structure the VCL is concerned with is traditionally known as a Slice – a collection of coded macroblocks in scan order.
- The Network Adaptation Layer (NAL), on the other hand, is responsible for the encapsulation of the coded slices into transport entities of the network, e.g. into packets, or into a start-code delimited bit stream.

Neither the VCL nor the NAL are transport or media unaware in the sense used in the MPEG-4 concept, see [2] for tutorial information. The VCL, for example, needs knowledge about the

constraints of the transport such as the MTU size (to appropriately select the slice size) or the error characteristics (to appropriately select the error resilience strength). The NAL, on the other hand, needs for example to know what types of symbols are included in the VCL bit string (to unevenly protect the more important bit strings or to determine the transmission timing).

Since there are many different networks and multiplex environments with very different characteristics, several different NAL specifications will emerge over time. So far, a number of mechanisms have been identified that are shared between all NALs: the Parameter Set concept, the NAL Packet types and some others. In addition, there are parts in the NAL specification that are clearly dependent on the network/multiplex environment. With respect to these network-dependent NAL parts, this paper focuses on the NAL variant for the transport of H.26L over best effort IP networks [3], with the most widely used protocol structure for all real-time multimedia over IP: UDP [4] on the transport layer, and RTP [5] with an appropriate payload specification [6] on the application layer. This paper assumes that the reader is familiar with the characteristics of such an environment, and it can be summarized as follows:

- Packets are transmitted bit error free.
- Packets can get lost.
- The loss probability depends on too many factors to be pre-determined and can be as high as 20% or more.
- The MTU size is network dependent but often in the neighborhood of 1500 bytes.

Also, due to space constraints, this paper does not describe the VCL in any detail – only this one-paragraph overview will be given. The VCL operates very similar to the traditional video coding standards – it is based on a hybrid of inter picture prediction with motion compensation and the transform coding of the residual. The macroblock (MB) size is 16x16 (luminance) pixel, but the basic block size is 4x4 pixel, and motion vectors (MVs) can be coded for a variety of macroblock modes, including one that transmits one MV for each 4x4 block. Macroblock types include intra and inter with the commonly understood meaning; and there are a number of MB types to support bi-directionally predicted slices. As for the error resilience tools the VCL relies on intra macroblock refresh whereby the mode selection is performed in a very sophisticated way in the test model [7]. Furthermore, the VCL supports (by definition, as it is not concerned with whole pictures) picture segmentation, and data partitioning. The latter is a shared functionality between VCL and NAL and is described in more detail in the following section.

This paper is organized in 5 sections. Section 2 provides some insight of the network-independent parts of the NAL, and in particular the Parameter Set concept. Section 3 describes the RTP specific part of the NAL. In section 4 a few simulation results are presented that show five different applications of the error resilience tools of the VCL and the NAL for the same video sequence and error patterns. Section 5, finally, concludes this paper.

## **2. Network independent NAL properties**

This section describes the properties of the network-independent features of the NAL as seen in the encoder. In the decoder, the processes occur in the reverse order.

### **2.1. VCL to NAL interface**

As mentioned before, the interface between the VCL and the NAL is at the Slice level. A Slice is defined as a concatenation of the entropy coded symbols of an integer number of macroblocks (MBs) in scan order. Each Slice has an associated header, which the VCL makes available to the NAL in the form of an implementation-dependent data structure. Structure elements include:

- Picture ID: a number that can be used to associate slices to pictures, similar to the temporal reference of older video coding standards. In H.26L the Picture ID does not necessarily carry timing information in the sense of a Temporal Reference (TR), and the numbering space can be selected at the sequence level (see the discussion of the Parameter Set concept below).
- Parameter Set: the picture-level (and higher) parameters used to code the slice.
- Macroblock Address: The spatial address of the first MB in the slice. This parameter is in most NALs coded in two codewords that represent the X and the Y dimensions separately as it is more efficient when UVLC codes are used.
- Slice Type: e.g. intra slice, inter slice, bi-directionally predicted slice.
- Entropy Coding: the entropy coding method used to code the slice. H.26L contains two entropy coding methods: Universal Variable Length Codes (UVLC), and Context Adaptive Binary Arithmetic Coding (CABAC). For the NAL the used entropy coding method is transparent.
- Others: a number of additional parameters, some of which are optional.

In addition to the slice header, the slice consists of three bit strings that carry the coded macroblock data. When not using data partitioning, the first bit string includes all symbols of the slice in an order very similar to traditional video coding standards, and the second and third bit strings are empty. When in data partitioning mode, the three bit strings are also called partitions. The first partition contains the header symbols of all coded MBs, the second partition the intra Coded Block Patterns (CBPs) and coefficients, and the third partition the inter CBPs and coefficients. For the reproduction an order of importance of the partitions can be observed in this scheme:

- The header partition is the most important one, because, without it, the coefficient symbols cannot be used. Also, the header symbols have a value of their own, e.g. they can be employed to make error concealment much more efficient by applying the motion vectors even in the absence of the corresponding texture data, see section 4 experiment 5 for results that verify the efficiency of this approach.
- The Intra partition is next on a scale of importance. While the Intra partition necessarily needs the symbols of the header partition for decoding, the resynchronization property of intra information is essential to combat the drift observed in predictive coding schemes that operate in error prone environments.
- Finally, the Inter partition contains the least important symbols.

The network specific part of the NAL can respond to the importance of the different partitions by unevenly protecting them. Such protection can be a function of the network, but can also be done on the application layer, e.g. by using packet duplication or a packet-based FEC [8].

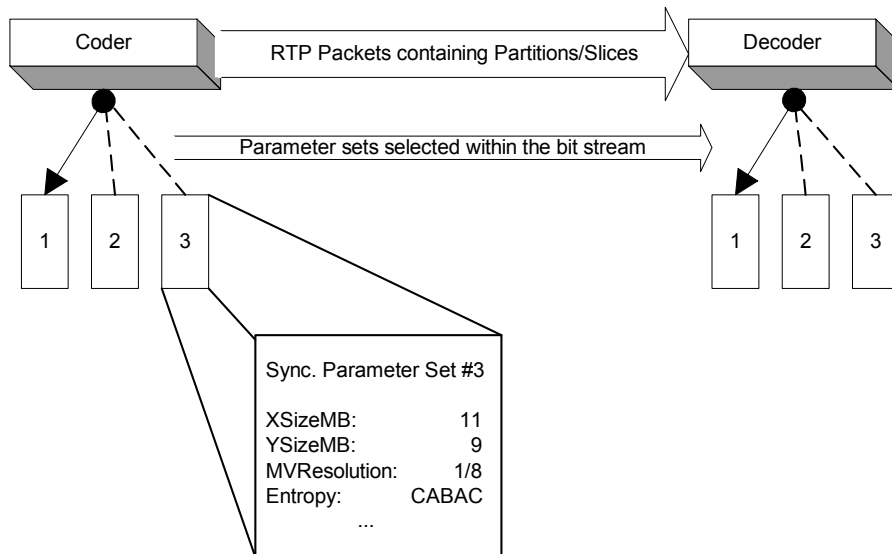
## 2.2. The Parameter Set concept

All video coding standards include some form of picture segmentation that is similar to the slice concept, and the rationale behind this inclusion is normally to allow for the adaptation to the MTU size. However, the traditional video coding standards do not keep the slices self-contained in the sense that they can be decoded independently from other picture information. More specifically, there are data structures such as the picture, or group-of-picture header, that are relevant for more than one slice, but carried only as part of one single slice (the first slice of a picture or GOP). Conceptually, while there are sometimes ways to conceal the missing headers, this header containing slice is critical for the decoding of the whole picture/GOP: if the critical slice gets lost, the remaining slices cannot be decoded. The standards reply to this problem by

allowing the carrying of header information redundantly in other slices as well, e.g. through MPEG-4's HEC concept [9], the header duplication feature of H.263 Annex W [10], or the similar mechanism in RFC 2429 [11]. However, such mechanisms have a negative impact on the coding efficiency and are architecturally not very clean. H.26L addresses the problem of lost picture headers in a much more elegant way by the introduction of the Parameter Set concept, which is described in this section.

The main observation that lead to the development of the Parameter Set concept is that picture and higher header parameters, with the exception of a few items such as the TR and the picture type, are normally quasi static over the lifetime of a session. In H.26L, those parameters that vary from picture to picture, for example the TR or the picture type, are moved to the slice layer. This ensures that slices are independently decodable and self-contained. All other information, such as the conformance point of the H.26L stream (be it called a profile/level combination or expressed in the use of optional modes) is conceptually conveyed out of band.

Traditionally, the picture header is synchronous to the bit stream, meaning that any change to picture parameters affect all slices and (when relative picture header updates are part of the standard as in H.263 and MPEG-4) pictures henceforth. Since the synchronization of out-of-band mechanisms with the slice stream of H.26L would be very difficult if not impossible, H.26L allows each slice to identify one of several "picture headers" that may have been sent a long time ago. These "picture headers" are called Parameter Sets, because that is what their nature really is – they are not bound to a certain picture (or GOP), they simply define the picture-global parameters the decoder has to assume when decoding a slice. Each slice header contains in its Parameter Set structure member an integer that references the to be applied parameter set. Figure 1 depicts this relationship:



**Figure 1: The Parameter Set concept**

Decoupling the transmission of the Parameter Sets from the video stream has several advantages:

- In many applications with powerful control protocols, such as video-conferencing, the establishment of common Parameter Sets between encoder and decoder may be a function of the capability exchange.
- Parameter Set updates can be sent using a reliable control protocol that ensures that the critical information is received unharmed.
- Since such control protocols usually include acknowledgement, the encoder is informed precisely about the earliest time it can rely on an updated Parameter Set.
- Some applications, e.g. digital TV, may have such a limited amount of configurable features that there may be simply no need to transmit Parameter Sets at all – they may be pre-configured in the ROM of the encoder and the decoder.

The only disadvantage that was so far discovered is that an encoder cannot react quickly to changing content by applying different coding modes on a per picture basis. Implementation experience, however, tells us that few (if any) encoders actually implement such a mechanism. However, in order to support even such sophisticated encoders, some network specific NALs may include an additional in-band transmission tool for updating the Parameter Sets, through which the traditional picture header mechanism can be emulated. These Parameter Set Update Packets (PUPs) are, of course, subject to the same error characteristics as the rest of the packet sequences. Because they carry vital information for the decoding of many other packets, they need to receive a very high level of protection. In IP networks this can be achieved, for example, by sending PUPs multiple times to ensure that at least one copy arrives at the receiver. Other application layer protection schemes can, of course, be used as well. See the discussion on the PUP-NALPs, below.

The number of sequence, GOP, and picture parameters necessary for H.26L is, at the current state of standardization, a surprisingly short list:

- Picture Size (in X, and Y dimensions, measured in units of macroblocks),
- Display Window (in X, and Y dimensions, measure in units of pixels),
- Entropy Coding type (UVLC or CABAC),
- Motion Vector Resolution ( $1/4^{\text{th}}$  pel or  $1/8^{\text{th}}$  pel)
- Modulo Value for the Picture ID
- Pixel Aspect Ratio

The list is so short, because, so far, no “optional” coding tools in the sense of H.263, except the different entropy coding schemes and the different MV resolution, have been identified.

Since the Parameter Set updates are conveyed out of band, no specific syntax for their coding is currently defined. Control protocols, as the preferred way for the transmission of Parameter Sets, have their own syntax constraints, which should not be questioned by the H.26L specification. However, if a network-specific NAL allows to transmit the Parameter Set updates in-band, then syntax has to be defined.

### **2.3. Basic Network Adaptation Layer Packets (NALPs)**

As mentioned before, the VCL delivers to the NAL a data structure containing the slice parameters and the bit strings of the coded MBs. Furthermore, the encoder may want to send so-called supplementary enhancement information (SEI) synchronous with the video stream. SEI carries information that is helpful for the user experience of the reproduced video, but does not affect the decoding process itself and, particularly, the handling of the reference pictures. There may also be a need for transmitting Parameter Set updates in-band. For all these mechanisms, a syntax needs to be defined. This syntax could, theoretically, be network dependent. However, so

far, the JVT group assumes the generalized syntax described here to be applicable for all networks.

The main observation of this high level syntax is that the transmission of video always was packet-based. Even in the oldest video coding standards such as H.261 [12], GOBs and pictures were separated by start codes. Clearly, such framing mechanisms resemble an archaic way of forming a (bit oriented) packet structure for the coded video.

If coded video is de-facto always packet based, one can as well make use of this property in an intelligent way, in order to facilitate gateway design, transcoding, packetization for true packet networks, and uneven protection of important bit strings. This is the rationale behind the introduction of the Network Adaptation Layer packets.

All NALPs consist of a one-byte header, and a variable number of payload bytes. The structure of the payload depends on the NALP type. The header is currently defined in a table (to facilitate the design of an MPEG-2 transport NAL [6]), but can conceptually be seen as a structure with two elements:

- NALP-type, an integer that identifies the type of the NALP, see below
- Error Indication, a bit that is set if the payload should be suspected to include bit errors, and cleared otherwise. This bit may, for example, be set by wireless to wireline gateways that have identified a bit error in the NALP by means of a checksum that is part of the wireless NAL. Doing so, the decoder is advised to only attempt the decoding of the NALP if it is bit error tolerant.

So far, six different basic NALP types are defined:

- Single Slice. This NALP carries as its payload a complete Slice including the UVLC coded Slice header and the coded MBs.
- DPA: Carries a full Slice header, a Slice ID, and the bit string of the header data partition. The Slice ID is necessary to associate the following coefficient partitions with the header partition.
- DPB: Consists of Picture ID, Slice ID, and the Intra coefficient partition.
- DPC: Picture ID, Slice ID, and the Inter coefficient partition
- SEI: Supplementary Enhancement Information NALP, as discussed above. For the precise syntax of this NALP please see [13]
- PUP: Parameter Update Packet. A NALP that allows to transmit in-band updates of one or more Parameter Sets. Currently, a SDP-like syntax [14] is assumed – however, the author could envision the definition of a more efficient syntax as well. See [15] for examples how to use PUPs.

In addition to these basic NALP types, a seventh NALP is defined that allows to multiplex basic NALPs together. This Compound NALP is described next.

## 2.4. Compound NALP

One of the key applications of H.26L seems to be the transmission of video in 3GPP wireless environments. Here, the wireless part of the network operates often with MTU sizes around 100 bytes, whereas in the wireline core network, MTU sizes of 1500 bytes are normal. Similarly, the overhead per packet in the wireless case is only a few bytes, whereas in the wireline case more than 40 bytes have to be assumed. In order to avoid transcoding (which adds delay and computational complexity), it would be advisable to code basic NALPs according to the smaller MTU size to avoid fragmentation and increase loss rates. This, however, leads to a remarkably bad payload/overhead ratio in the wireline part of the network. A maximum of 100 bytes payload

would require a minimum of 40 bytes header information. One way out of this dilemma is to support application layer multiplexing through the Compound NALP.

There are also other very compelling reasons for the introduction of Compound NALPs, e.g. the transmission of DPA NALPs belonging to several slices (or even pictures) in one transport packet that receives heavy protection, the combining of DPAs and DPBs in Intra Slices when using Data Partitioning, or the use of Slice Interleaving (see section 4). Space constraints disallow the author to discuss those topics further, but some details can be found in [15].

A Compound NALP consists of the usual one byte NALP header, and a number of sub-packets. Each sub-packet consists of a sixteen bit integer indicating the length of the sub-packet, and one simple NALP. This trivial packet structure can be parsed even by the most basic gateways in order to pack or unpack Compound packets according to the application's needs, at a computational complexity level that is by orders of magnitudes smaller than other mechanisms MTU-size changing gateways typically employ (i.e. IP header compression).

The decision which basic NALPs to carry in one Compound packet is not a trivial one. The simplest possible algorithm would be to collect basic NALPs belonging to only one single picture in their order of production, until the generated Compound packet reaches the MTU size. However, more intelligent schemes can also be identified, such as slice interleaving as discussed below in section 4. Finally, delay-insensitive applications such as streaming can packetize basic NALPs belonging to more than one picture together and thus greatly reduce the header overhead for low bit rate streaming applications.

### ***3. RTP-specific parts of the NAL***

When using RTP as the transport for media, all but the most basic media codecs require the use of an adaptation scheme called the RTP payload specification. The RTP payload specification [6] for H.26L is currently in an early stage of standardization, although many of the mechanisms described there are already implemented and verified independently. Some of the results reported in section 4 below belong to this verification as well.

For most video coding schemes, the RTP packetization describes three major aspects:

- The Fragmentation Rules, which are rules that define how to split the video bit stream into packets.
- The transmission of redundant header information to make RTP packets less dependent on each other, e.g. by header duplication mechanisms in the payload header.
- The use of the RTP header fields, especially the RTP time stamp and the Marker bit.

For H.26L, the first two aspects are already handled in the generic NAL with an acceptable quality. The packetization schemes spells out some constraints about the composition of Compound packets, but other than that the fragmentation rules are very simple: one NALP per RTP packet. The transmission of redundant header information is unnecessary, because due to the Parameter Set concept no headers exist that need to be protected.

The main issue in the draft payload specification is hence concerned with the definition of the RTP header fields. Here, the draft as it stands at the time of publication of this paper, is relatively vague, making a proposal that is not completely in-line with other modern RTP payload specifications. The discussion is ongoing, and completely different solutions may be necessary and found. For this reason, the author believes a further discussion in this paper would not be productive, and refrains from doing so.

## 4. Experimental Results

### 4.1. Environment

This section contains the results of some simulations performed with the current release of the TML software, JM 1.0, and a set of network simulation tools that are generally used by JVT. In addition, a few small stand-alone tools for tasks such as the duplication of header partitions (to improve their packet loss rate) or to implement compound packets for interleaved packetization were required. Tests were performed according to the generally used Common Conditions for wireline IP environments [16] and using the IP error patterns of [17]. Due to space constraints, results are reported only for a single sequence: Foreman, in QCIF format, with a gross target bit rate of 64 kbit/s (including any IP/UDP/RTP/NAL overhead) and a frame rate of 7.5 frames per second (fps).

A total of five experiments were performed.

1. One picture – one packet, without any error resilience tool applied. This experiment shows the best performance in error free tests as no redundancy is introduced for error resilience, and the worst in error prone tests. Many academic publications use this (unfair) setting in order to prove the efficiency of a new error resilience tool. Here, it is only included to show how ridiculous such a comparison is.
2. One picture – one packet, with appropriate intra refresh based on the loss-aware R/D optimization scheme reported in [7].
3. Three equally sized slices per picture, with appropriate intra refresh based on the loss-aware R/D optimization scheme.
4. Nine slices per picture that are packetized into two compound packets using the Slice interleaving scheme as described in RFC 2429 [11]. This scheme packetizes all even numbered rows of MBs in one RTP packet, and all odd rows in another packet. Such a scheme is widely used in videoconferencing products based on H.263. Intra is used as in the above experiments.
5. One Slice with three Partitions per picture, no Intra refresh, no rate-distortion optimization, appropriate packet duplication rates to protect the Header Partition, best effort for the coefficients.

Experiments 3, 4, and 5 shared the same sophisticated error concealment scheme described in [13]. That scheme is a hybrid of a temporal MV concealment (where the motion vectors are “guessed” from surrounding motion vectors) and a relatively weak spatial interpolation concealment. Experiments 1 and 2 could not use any error concealment, because the decoder has no information about the lost picture(s) whatsoever which it could employ as a hint for heuristic means.

Figure 2 plots the results of all five experiments for the Foreman sequence at the packet loss rates of 0%, 3%, 5%, 10%, and 20%.

For the diagrams a regular PSNR measurement was used. The PSNR was only calculated for the pictures that were actually reconstructed (in whole or in part with error concealment). No penalty was introduced for lost pictures. If such were the case, one could expect that the quality for the Experiments 1, 2 and 4 would have been significantly worse than indicated in the diagrams, because in all those experiments a relatively large number of complete pictures would have gotten lost, giving no hint to the decoder that they existed. In Experiment 5, the Header Partition was protected well enough that here few, if any, pictures got lost.

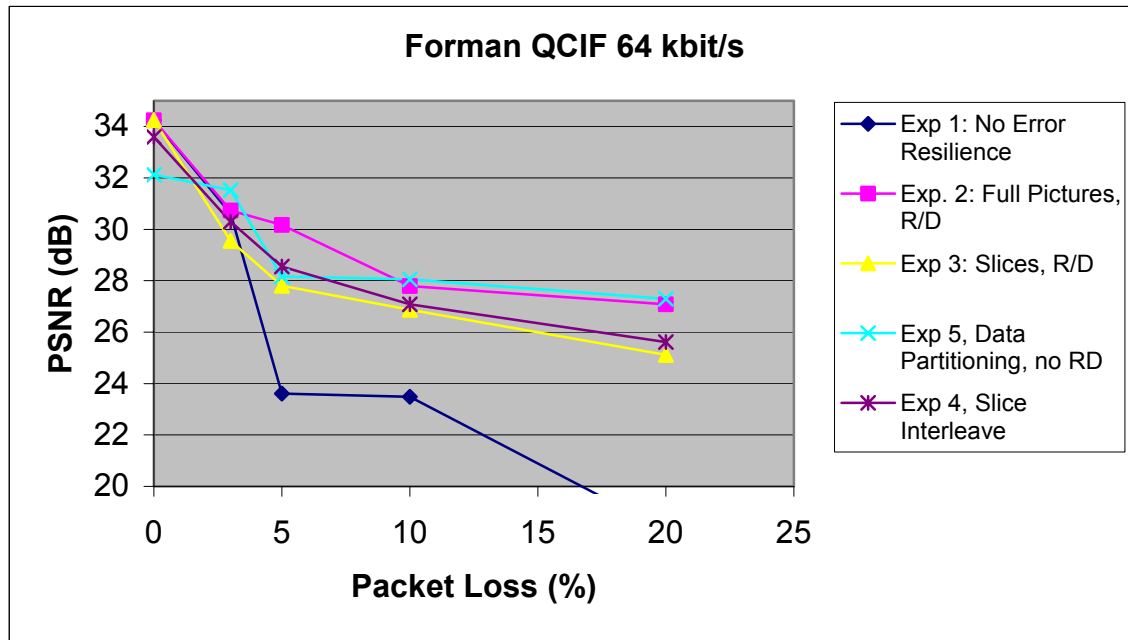


Figure 2: Experimental Results

## 4.2. Discussion of the Results

### 1) Experiment 1: no error resilience

Obviously, the results for Experiment 1, where no bits are spent for error resilience purposes, are the best in the error free and the worst in all error prone test runs. It is fair to say that at packet loss rates above 3% unprotected H.26L video becomes unusable. This is in line with research on other video coding schemes that employ inter picture prediction.

### 2) Experiment 2: one picture, one packet

The one-picture, one-packet approach of Experiment 2 and for the Foreman sequence yields very good results. However, the picture loss rate is statistically the same as the packet loss rate. This, in conjunction with the low frame rate of only 7.5 fps leads to annoying picture freezes especially when somewhat longer error bursts occur at packet loss rates of 10% and 20%. The loss-aware rate-distortion optimization process selects enough intra MBs that the quality degradation is minimal, even at higher packet loss rates. However, this is accomplished by using a very high amount of intra MBs of significant size, which leads to a coarser quantizer value and, hence less spatial detail. The IP/UDP/RTP header overhead is kept to the theoretical minimum for low-delay applications.

### 3) Experiment 3: Slices

In Experiment 3 it was tried to adjust the IP/UDP/RTP header overhead to a similar amount as used by the Experiments 4 and 5. This has the welcoming side effect that the picture loss rate is minimized, because it is very likely that at least one slice of every coded picture is received. Three packets per Picture were used. Due to the additional header overhead fewer bits were available to code video, and, hence, a coarse quantizer had to be selected to stay within the bit rate requirements. For the Foreman sequence, this led to unfavorable PSNR results. However, subjectively, and in contrast to the objective PSNR results, Experiment 3 is superior to Experiment 2 due to the constant high frame rate.

#### **4) Experiment 4: Slice Interleaving**

Experiment 4 used Slice Interleaving, a technique that has turned out to be successful for other video coding standards, and is used commercially in several video telephony systems. At least two packets per picture were used. The penalty due to the increased header overhead, when compared to Experiment 2, led to a slightly inferior objective result, while, subjectively, the constant reproduced frame rate made a better impression than Experiment 2.

#### **5) Experiment 5: Data Partitioning**

Experiment 5 must be distinguished from the other experiments because it does not employ any intra refresh for error resilience purposes. It relies completely on the superior error concealment mechanisms that are available when it can be made sure (or almost sure) that at least the vital information of the header partition arrives. This is achieved by sending the RTP packet with the header partition twice (for the 3% error rate case) or 3 times (for the 5%, 10%, and 20% error rate cases). Of course, the quantizer is adapted so that the complete packet stream, including the multiple header partitions, fits into the bit rate budget. No loss-aware Rate Distortion Optimization was used for this Experiment. Hence, the computational encoding complexity was significantly lower than in the other experiments. With the JM1.1 reference software, the encoding speed for Experiment 5 was more than twice higher than those for the other experiments.

Considering this, the results were impressive; the scheme outperformed all other experiments both subjectively and objectively. However, the results were not unexpected – old research based on earlier versions of the H.26L reference software were able to produce similar results, see [18]. Undoubtedly, the performance of Experiment 5 could be further increased when using an optimized intra refresh method for this scheme, but thus far time constraints have disallowed the implementation of such a mechanism.

### **5. Conclusion**

In this paper, the current state of standardization of the Network Adaptation Layer of the JVT video codec and its transport over RTP was outlined. The NAL should not be seen as an abstraction layer that hides network features, but rather as the often missed tool that allow the video codec to change the bit stream appearance so that it suit a network. The main new features of the NAL are the Parameter Set concept that decouples the (synchronous) video stream from the (asynchronous) transmission of picture header information, and the introduction of Network Adaptation Layer packets. The RTP packetization scheme is currently in a too early state of development that it could be described in detail here. However, based on the current NAL and RTP packetization specifications, an implementation emerged. Based on this implementation, quite impressive simulation results could be reported in this paper.

### **References**

---

- [1] G. Sullivan, VCEG, MPEG, “Terms of Reference for a Joint Project between ITU-T Q.6/SG16 and ISO/IEC JTC 1/SC 29/WG 11 for the Development of a new Video Coding Recommendation and International Standard”, VCEG-O54, available from [http://standard.pictel.com/ftp/video-site/0112\\_Pat/VCEG-O54.doc](http://standard.pictel.com/ftp/video-site/0112_Pat/VCEG-O54.doc), December 2001
- [2] ISO/IEC JTC1/SC 29/WG 11, “Overview of the MPEG-4 Standard (V.18),” ISO/IEC JTC1/SC 29/WG 11 N4030, March, 2001. Current version available at <http://www.cse.it/mpeg/standards/mpeg-4/mpeg-4.htm>.
- [3] J. Postel, “Internet Protocol”, RFC 791, September 1981
- [4] J. Postel, “User Datagram Protocol”, RFC 768, August 1980

- 
- [5] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996
  - [6] S. Wenger, T. Stockhammer, M. Hannuksela, "RTP payload Format for JVT Video", draft-wenger-avt-rtp-jvt-00.txt, Internet Draft, Work in Progress, February 2002
  - [7] T. Stockhammer, D. Kontopodis, T. Wiegand, "Rate-Distortion Optimization for H.26L Video Coding in Packet Loss Environment", Proc. Packet Video Workshop 2002
  - [8] J. Rosenberg, H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction", RFC 2733, December 1999
  - [9] ISO/IEC JTC1, "Coding of audio-visual objects – Part 2: Visual," ISO/IEC 14496-2 (MPEG-4 visual version 1), April 1999; Amendment 1 (version 2), February, 2000; Amendment 4 (streaming profile), January, 2001.
  - [10] ITU-T Recommendation H.263 Annex W (2000), available from the ITU-T, <http://www.itu.int>
  - [11] C. Bormann, L. Cline, G. Deisher, T. Gardos, C. Maciocco, D. Newell, J. Ott, G. Sullivan, S. Wenger, C. Zhu, "RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)", RFC2429, October 1998
  - [12] ITU-T Recommendation H.261, "Video codec for audiovisual services at p x 64 kbit/s", 1993, available from the ITU-T, <http://www.itu.int>
  - [13] H.26L working draft, the most up-to-date version is available from <http://standard.pictel.com/ftp/video-site/h26l/jwdxx.doc>, where xx is the version number
  - [14] M. Handley, V. Jacobson, "SDP: Session Description Protocol", RFC 2326, April 1998
  - [15] S. Wenger, T. Stockhammer, "H.26L over IP and H.324 Framework", VCEG-N52, available from [http://standard.pictel.com/ftp/video-site/0109\\_San/VCEG-N52.doc](http://standard.pictel.com/ftp/video-site/0109_San/VCEG-N52.doc), September 2001
  - [16] S. Wenger, "Common Conditions for wire-line, low delay IP/UDP/RTP packet loss resilient testing", VCEG-N79r1, available from [http://standard.pictel.com/ftp/video-site/0109\\_San/VCEG-N79r1.doc](http://standard.pictel.com/ftp/video-site/0109_San/VCEG-N79r1.doc), September 2001
  - [17] S. Wenger, "Internet Error Patterns", VCEG-N16r1.zip, available from [http://standard.pictel.com/ftp/video-site/9910\\_Red/Q15-I16r1.zip](http://standard.pictel.com/ftp/video-site/9910_Red/Q15-I16r1.zip), October 1999
  - [18] S. Wenger, "A High Level Syntax for H.26L: First Results", IEEE International Conference on Visual Communication and Image Processing, VCIP 2000, Perth, Australia, 2000