Review of Error Resilient Coding Techniques for Real-Time Video Communications

Yao Wang, Stephan Wenger, Jiangtao Wen, and Aggelos K. Katsaggelos[1]

Abstract **–**In this paper we review error resilience techniques for real-time video transport over unreliable networks. Topics covered include an introduction to today's protocol and network environments and their characteristics, encoder error resilience tools, decoder error concealment techniques, as well as techniques that require cooperation between encoder, decoder and the network. We provide a review of general principles of these techniques as well as specific implementations adopted by the H.263 and MPEG-4 video coding standards. The majority of the paper is devoted to the techniques developed for block-based hybrid coders using motion-compensated prediction and transform coding. A separate section covers error resilience techniques for shape coding in MPEG-4.

I. Introduction

A. Error Resilience in Video Communications: Importance and Approach

A video communications system typically involves five steps, as shown in Figure 1. The video is first compressed by a video encoder to reduce the data rate and the compressed bit stream is then segmented into fixed or variable length packets and multiplexed with other data types, such as audio. The packets

[1] Y. Wang is with Polytechnic University, Brooklyn, NY 11201, USA (e-mail: yao@vision.poly.edu). S. Wenger is with Technische Universität Berlin, Sekr. FR 6-3, Franklinstrasse 28-29, D-10587 Berlin, Germany (e-mail: stewe@cs.tu-berlin.de). J. Wen is with PacketVideo Corp., San Diego, CA 92121, USA (email: gwen@packetvideo.com). A. K. Katsaggelos is with Northwestern University, Dept. of ECE, Evanston, IL 60208, USA (e-mail: aggk@ece.nwu.edu)

might be sent directly over the network, if the network guaranties bit error free transmission. Otherwise, they usually undergo a channel encoding stage, typically using forward error correction (FEC), to protect them from transmission errors. At the receiver end, the received packets are FEC decoded and unpacked, and the resulting bitstream is then input to the video decoder to reconstruct the original video. In practice, many applications embed packetization and channel encoding in the source coder as an adaptation layer to the network.

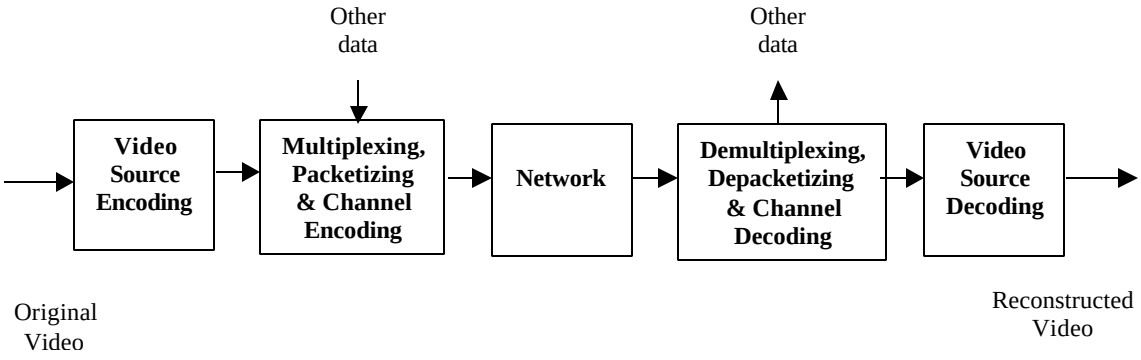

Figure 1. A typical video communication system.

Unless a dedicated link that can provide a guaranteed quality of service (QoS) is available between the source and destination, data packets may be lost or corrupted, due to either traffic congestion or bit errors due to impairment of the physical channels. Such is the case, for example, with the current Internet and wireless networks. In such situations, error-free delivery of data packets can only be achieved by allowing retransmission of lost or damaged packets, through the mechanisms such as Automatic Repeat Request (ARQ). Such retransmission however may incur delays that are unacceptable for certain real-time applications. Broadcast applications prevent the use of retransmission algorithms completely due to network flooding considerations. Therefore, it is important to devise video encoding/decoding schemes that can make the compressed bitstream resilient to transmission errors. It is also prudent to design proper interfacing mechanisms between the codec (encoder and decoder) and the network, so that the codec can adjust its operations based on the network conditions.

Error control in video communications is very challenging for several reasons. First, compressed video streams are very sensitive to transmission errors because of the use of predictive coding and variable length coding (VLC) by the source coder. Due to the use of spatio-temporal prediction, a single erroneously recovered sample can lead to errors in the following samples in the same and following

frames. Likewise, because of the use of VLC, a single bit error can cause the decoder to lose synchronization, so that even correctly received following bits become useless. Figure 2 shows a typical example of reconstructed frames in the presence of packet loss. Secondly, the video source and the network conditions are typically time-varying, so that it is hard or impossible to derive an "optimal" solution based on some statistical models of the source and network. Finally, a video source has a very high data rate, therefore the encoder/decoder operations cannot be overly complex, especially for real-time applications.

Figure 2: Effect of transmission errors to a compressed video stream using the H.263 standard for a selected frame. Upper left: no transmission errors were present and the picture quality is as high as the bitrate allows; Upper right: 3% packet loss; lower left: 5% packet loss; lower right: 10% packet loss.

To make the compressed bit stream resilient to transmission errors, one must add redundancy into the stream, so that it is possible to detect and correct errors. Such redundancy can be added in either the source or channel coder. The classical Shannon information theory states that one can separately design the source and channel coders, to achieve error-free delivery of a compressed bit stream, as long as the source is represented by a rate below the channel capacity. Therefore, the source coder should compress a source as much as possible (to below the channel capacity) for a specified distortion, and then the channel coder can add redundancy through FEC to the compressed stream to enable the correction of transmission errors. However, such ideal error-free delivery can be achieved only with infinite delays in implementing

FEC and are not acceptable in practice. Therefore, joint source and channel coding is often a more viable scheme, which allocates a total amount of redundancy between the source and channel coding. All the *error resilient encoding* methods essentially work under this premise, and intentionally make the source coder less efficient than it can be, so that the erroneous or missing bits in a compressed stream will not have a disastrous effect in the reconstructed video quality. This is usually accomplished by carefully designing both the predictive coding loop and the variable length coder, to limit the extent of error propagation.

Even when an image sample or a block of samples are missing due to transmission errors, the decoder can try to estimate them based on surrounding received samples, by making use of inherent correlation among spatially and temporally adjacent samples. Such techniques are known as *error concealment* techniques. Again, this is possible because real source coders do not completely eliminate the redundancy in a signal in the encoding process. For example, the encoder typically periodically restarts the prediction process, to limit the effect of error propagation. A consequence of this intentional deficiency of the encoder is that a transmission error may affect only a middle part of a frame, which can then be estimated by spatial and temporal interpolation. To facilitate decoder error concealment, the compressed data for adjacent samples or blocks may also be packetized in an interleaved manner, to increase the likelihood that a damaged region is surrounded by undamaged regions. Error concealment has, in contrast to error resilient source coding, the advantage of not employing any additional bitrate, but adds computational complexity at the decoder.

Finally, for the embedded redundancy in the source coder to be useful, and to facilitate error concealment in the decoder, the codec and the network transmission protocol must cooperate with each other. For example, if the bitstream is such that some bits are more important than others, then the important part should be assigned a more stringent set of QoS parameters for delivery over a network. To suppress error propagation, the network may also provide a feedback channel, so that the encoder knows which part of the reconstructed signal at the decoder is damaged, and do not use this part for prediction of future samples.

To summarize, mechanisms devised for combating transmission errors can be categorized into three groups: i) those introduced at the source and channel encoder, to make the bit-stream more resilient to potential errors; ii) those invoked at the decoder upon detection of errors, to conceal the effect of errors; and iii) those which require interactions between the source encoder and decoder, so that the encoder can

adapt its operations based on the loss conditions detected at the decoder. In this paper, we refer to all of them as error resilience (ER) techniques.

The purpose of this paper is to review such techniques in general, as well as their specific implementations in two recent video coding standards, H.263 and MPEG-4. For other recent reviews see [1,2,3,4] and references therein. Sections II-IV will focus on video coders using block-based temporal prediction and transform coding, since such coders are presently the most practical and effective ones and have been adopted in all international video coding standards. Section V will review ER tools developed for shape coding in MPEG-4.

Before moving onto the review of ER techniques, in the remainder of this section, we briefly describe in Sec. I.B the block-based hybrid video coding method, for the benefit of the readers who are not familiar with this coding paradigm, and for introducing necessary terminology. We also review in Sec. I.C characteristics of practical networks and requirements for different applications. These are important factors to consider, because the necessity for error control and the effectiveness of a technique depends on the type of applications as well as the underlying network protocols.

B. Block-Based Hybrid Video Coding Framework

Figure 3 shows the key steps in this coding paradigm. As illustrated, each video frame is divided into blocks of a fixed size and each block is more or less processed independently, hence the name "block-based". The word "hybrid" means that each block is coded using a combination of motion-compensated temporal prediction and transform coding. That is, a block is first predicted from a matching block in a previously coded reference frame. The estimation of the location of the best matching block is known as *motion estimation*, and the displacement between the current block and the matching block is represented by the *motion vector* (MV). The process of predicting a block based on the MV is called *motion compensation.* The use of prediction is motivated by the fact that a current block is usually similar to a previous block, and that it is wasteful of bits to specify the pixel values in the current block directly. Instead, the prediction error block is specified, by converting it using the *discrete cosine transform* (DCT), quantizing the resulting coefficients, and converting them into binary codewords *using variable length coding* (VLC). The purpose of DCT is to reduce the spatial correlation between adjacent error pixels, and to compact the energy of the error pixels into a few coefficients. Because many high-frequency coefficients are zero after quantization, VLC is accomplished by a *runlength coding* method, which orders the coefficients into a one-dimensional array using the so-called zig-zag scan so that the

low-frequency coefficients are put in front of the high-frequency coefficients. This way, the quantized coefficients are specified in terms of the non-zero values and the number of the preceding zeros. Different symbols, each corresponding to a pair of zero-runlength, and non-zero value, are coded using variable length codewords.

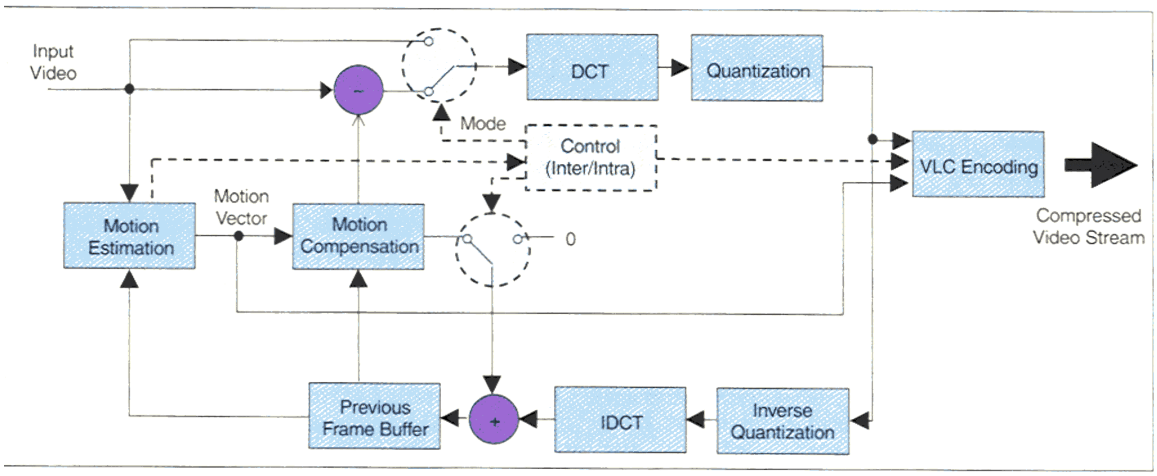

Figure 3 A typical Video Coder using block-based motion-compensated temporal prediction and transform coding. From Fig. 5 in [5].

The above discussion assumes that temporal prediction is successful, in that the prediction error block requires fewer bits to code than the original image block. This method of coding is called *P-mode*. When this is not the case, the original block will be coded directly using DCT and run length coding. This is known as *INTRA-mode*. Instead of using a single reference frame for prediction, bi-directional prediction can be used, which finds two best matching blocks, one in a previous frame and another in a following frame, and uses a weighted average of the two matches as the prediction for the current block. In this case, two MVs are associated with each block. This is known as the *B-mode*. Both P- and B-modes are generally referred to as *INTER-mode.* The mode information, the MVs, as well as other side information regarding picture format, block location, etc., are also coded using VLC.

In practice, the block size for motion estimation may not be the same as that used for transform coding. Typically, motion estimation is done on a larger block known as *macroblock* (MB), which is sub-divided into several blocks. For example, in most video coding standards, the MB size is 16x16 pels and the block size is 8x8 pels. The coding mode is decided at the MB level. Because MVs of adjacent MBs are usually similar, the MV of a current MB is predictively coded, using the MV of the previous MB for prediction.

Similarly, the DC coefficient of a block is predictively coded, with respect to the DC value of the previous block. In all discussed video coding standards, a number of MBs form a *Group of Blocks (GOB)* or a *Slice*, and several GOBs or Slices form a picture. Size and shape of GOBs and Slices differ among the various video coding standards and picture sizes, and can be tailored to the applications needs. Prediction of MVs and DC coefficients are usually restricted within the same GOB or Slice.

A frame may be coded entirely in INTRA-mode, and such a frame is called an *INTRA frame* or *INTRA picture.* This is used for encoding the first frame of a sequence. In applications employing high bitrate or with relaxed real-time constraints, INTRA frames are also used periodically to stop potential error propagation, and to enable random-access. Low latency applications cannot rely on this powerful means because INTRA frames are typically several times larger than any predicted frame. A *P-frame* uses only a past frame for prediction, and depending on the prediction accuracy, a MB can be coded in either INTRA or P-mode. Finally, a B-frame uses bi-directional prediction, and a MB in a B-frame can be coded in I-, P- or B-mode. A B-frame can only be coded after the surrounding INTRA- or P-frames are coded. All the techniques presented in this paper deal with error resilience in video coders using only INTRA- or P-mode. The word INTER frame is sometimes used to describe a P-frame. Error resilience for B-mode or any coding method using more than one reference frame is still a topic that needs further research.

C. Characteristics of Practical Networks and their Video Capable Applications

In this paper we focus on error control for video conveyed over current networks, using current protocol hierarchies. The combination of network and protocol characteristics allows us to describe the error characteristics of each combination, with which the video transmission process has to cope with. For all cases it is assumed that the application environment does not regularly allow retransmission of damaged or lost video data because of real-time constraints and/or broadcast transmission characteristics. This is, in today's environment, generally a valid assumption.

The International Telecommunication Union Telecommunication Standardization Sector (ITU-T) has produced a number of international standards (the H-series) for real-time digital multimedia communication. In addition, the Moving Pictures Experts Group (MPEG), an international standards committee, has produced a set of standards for audio and video compression. Formally MPEG is Working Group 11 of Subcommittee 29 of Joint Technical Committee 1 of the International Standards Organization (ISO) and the International Electrotechnical Commission (IEC). While MPEG-1 was developed for storage applications, MPEG-2 and MPEG-4 are used for applications requiring the

transmission of compressed digital video. Table 1 lists the target networks, the video coding and multiplex standards, and transmission characteristics, such as packet sizes and error rates, of these standards. Note that the visual and system parts of MPEG-2 were developed jointly by ITU-T and ISO/IEC (H.262 corresponds to MPEG-2 video, and H.222 corresponds to MPEG-2 systems).

Table 1: Standard Families for Video Transmission

Application and standard family	Multiplex protocol	Video coding standards used	Typical bitrate for video	Packet size	Error characteristics
ISDN Videophone (H.320)	H.221	H.261 and H.263	64 – 384 kbit/s	N/A	Error free
PSTN Videophone (H.324)	H.223	H.263	20 kbit/s	100 bytes	Very few bit errors and packet losses
Mobile Videophone (H.324 wireless)	H.223 w/ mobile extensions	H.263	10 – 300 kbit/s	100 bytes	BER=10E-3 to 10E-5, losses of H.223 packets
Videophone over Packet network (H.323)	H.225 / RTP/ UDP/ IP	H.261, H.263, MPEG-2	10 – 1000 kbit/s	<=1500 bytes	BER = 0, 0-30% packet losses
Cable/Satellite TV	H.222	MPEG-2	6 – 12 Mbit/s	N/A	Almost error free
Videoconferencing over 'Native' ATM (H.310, H.321)	H.222	MPEG-2	1 – 12 Mbit/s	53 bytes (ATM cell)	Almost error free

Currently, digital compressed video is regularly conveyed over the following networks and protocol hierarchies:

? ISDN (Integrated Services Digital Network) using ITU-T H.320: H.320 is the protocol hierarchy currently employed by the vast majority of video-conferencing and video telephony equipment [6]. The multiplex protocol H.221 [7] used by H.320 systems offers a bit oriented, practically error free video transmission channel with a fixed video bitrate. Beyond the mandatory INTRA MB refresh mechanisms of the video coding standards, no other ER tools are necessary or helpful. Being practically error free, this environment is not discussed any further in the rest of the paper.

? Cable-TV/Satellite channels using MPEG-2 transport streams [8]: This transport environment is employed by a vast majority of digital television applications, and is the most widely deployed environment for digital video as of today. Regardless of the underlying wireline or wireless physical layer, the channel coders and the MPEG-2 transport layer ensure an almost error free environment. No ER mechanisms beyond those mandatorily required by the standards are necessary. As with ISDN/H.320, there is no need for further discussion here.

? IP/UDP/RTP-based real-time transmissions on private IP networks or the Internet [9]: In this environment at least two high-level protocol architectures exist, namely H.323 [10] and "native" Internet announcement protocols such as SIP [11] and SDP [12]. Generally, the media transport is packet-based, and each media packet has an overhead of 40 bytes, typically yielding large data packets around 1500 bytes, to gain a reasonable payload/overhead relationship. The packets are transmitted bit-error free. The packet loss rate depends on the network conditions and can be as low as 0% in case of a highly over provisioned private IP network or as high as 30% and more for long distance connections during peak time on the Internet [13]. Through the feedback mechanisms of Real-Time Control Protocol (RTCP, part of the RTP specification) [9] it is possible to get information about the packet loss rates every few seconds, which allows for sender-based, adaptive use of ER tools. At the decoder, lost packets can be easily identified by the use of the RTP sequence number. In real-world systems this information is typically conveyed to the video decoder, which often uses it to enable decoder-based error concealment techniques. No further means for the location of errors are available or necessary.

? Internet streaming specific: Internet streaming applications such as the RealPlayer or similar products typically employ comparable protocol mechanisms like the ones discussed above. In addition, the application allows, due to its relaxed real-time demands where playout delays of several seconds are acceptable, for transport-based retransmission of missing or damaged parts of the media stream, thereby gaining an almost loss free environment. Therefore, there is little need for error resilient video coding beyond the mechanisms mandatory to the standards.

? H.324-based transmission over telephone network and wireline modems [14]: Systems conforming to these standards are available since 1996, but never gained market relevance due to their poor audio/video performance resulting from the very low available bitrate. In such systems, the tradeoff between bit error rate and bitrate of the modem link can be adjusted using the V.80 modem control mechanisms. Most systems train the modem path conservatively yielding reasonably low error rates

at some cost of bitrate, which is a necessity considering the problems arising from high bit error rates that the multiplex protocol H.223 (not using the 'mobile' extensions, see below) has [15]. Therefore, there is little need for error resilient video coding in this case.

In addition, there are a few networks and protocol environment combinations that are not yet regularly used, but deserve attention for various reasons specific to each combination:

? H.324/H.223 over wireless networks: This environment, often called the "mobile environment" is currently the most often discussed application for error resilient video coding, although the need for such coding is by no means clear yet[2]. Generally, any wireless link, especially when mobile stations are involved, which lead to sub-optimal antenna characteristics, has severe bit error characteristics. It is, however, similarly true, that, for real-world applications, quite sophisticated channel coders are used that reduce the bit error rates significantly. On top of these channel coders, real-world systems generally need some form of channel multiplexers that often include transport protocol functionality as well, thereby reducing the error rates further. A typical example for a practically error free wireless network was already mentioned above with the MPEG-2 transport/satellite combination. Many current proposals for wireless interactive multimedia communication employ H.223 and its "mobile extensions" as the transport/multiplex protocol on top of the bit-oriented channel. The mobile extensions form a hierarchy with five different levels, which allows for a scalable tradeoff between the robustness against bit errors of the multiplexer itself and the overhead incurred by that multiplexer. With the exception of the highest level, no improvement to the quality of the media transport is performed. Therefore, the video codec has to be able to cope with bit errors. The characteristics of these bit errors are typically expressed in the parameters bit error rate (BER) and

[2] There are currently at least three different proposals for video-capable communication over third generation wireless links under discussion: 1) the use of H.324 and its mobile extensions as the transport, 2) the use of IP/UDP/RTP-based transport employing H.323 type protocols for administration and 3) modified H.324-type transport with a different multiplex architecture. For cases 1) and 3), the discussed error characteristics and the means the video codec has to cope with those errors are valid. If alternative 2) is realized, then physical and link layer protocols have to be used that guarantee an almost bit error free environment, due to the characteristics of IP, UDP and RTP. In such a case, mobile communication could assume a bit-error free, but packet lossy link.

average burst length. Most research for video coding on top of the H.223 transport is performed with average burst lengths in the neighborhood of 16 bits, and BERs between 10E-3 and 10E-5. H.223 conveys media data including compressed video in the form of packets of variable size. Typical packet sizes are around 100 bytes to ensure good delay characteristics. If bit errors damage the protocol structures of H.223 beyond the repair facilities of the employed level (a condition known as multiplex errors), then whole packets can get lost as well. Therefore, video transmission on top of H.223 has to cope with packet losses. The location of errors in a system employing H.223 is a complex task. When using a H.223 level appropriate to the error characteristics of the link, then the multiplex errors occur only rarely and can be reliably detected by the multiplex protocol itself. The typical reaction of a H.223 implementation to a multiplex error is to drop the corrupted packet(s). In addition, H.223 conveys for video packets a Cyclic Redundancy Check (CRC) that allows the detection of bit errors in the payload of a video packet. Such packets can either be dropped, leading to a packet-lossy situation at the decoder, or the decoder can be informed by the transport hierarchy that a bit error is present in a packet.

? ATM cell-lossy networks: Research on video transmission over cell-lossy networks was several years ago one of the most prominent research topics in the signal processing community. Recently, fewer publications appear on this topic, mostly because the number of end-to-end ATM connections is far smaller than expected. Generally, cell loss can be considered as a sub-form of packet loss, whereby cells are extremely small packets. The mechanisms to cope with cell losses, however, are different, because it is not effective to add synchronization markers at the beginning of each cell, from a resilience/overhead tradeoff perspective. In this paper, we will not discuss cell lossy networks any more, mainly because of the large number of prior publications [16] and the limited number of applications.

? "Hypothetical" networks: A large number of publications discuss the behavior of a newly developed error resilience scheme in conjunction with a "hypothetical" network. Most often, bit error prone channels of over simplified characteristics (such as random bit errors) are used, under the assumption that the compressed video is transmitted directly over such a channel, without the use of any channel coder and/or multiplexer. In this paper we focus primarily on mechanisms that are supported by existing standards and are therefore utilized in real-world applications.

II. Review of Techniques for Error Resilience

In this section, we review general techniques that have been developed for error resilient video coding. Applications or special implementations of some of these tools in H.263 and MPEG-4 video coders are discussed in Sections III to V. As described in the introduction, ER techniques can be divided into three categories, depending on the role that the encoder, decoder, or the network layer plays in the process. We describe these in separate subsections. Our discussion assumes that video is coded using the block-based hybrid encoding framework described in Sec. I-B (cf. Figure 3). We will give only brief overviews of the techniques covered in [1], to leave space for more recent developments.

A. Error Resilient Encoding

In this approach, the encoder operates in such a way so that transmission errors on the coded bitstream will not adversely affect the decoder operation and lead to unacceptable distortions in the reconstructed video quality. Compared to coders that are optimized for coding efficiency, ER coders typically are less efficient in that they use more bits to obtain the same video quality in the absence of any transmission errors. These extra bits are called *redundancy* bits, and they are introduced to enhance the video quality when the bitstream is corrupted by transmission errors. The design goal in ER coders is to achieve a maximum gain in error resilience with the smallest amount of redundancy.

There are many ways to introduce redundancy in the bitstream. Some of the techniques help to prevent error propagation, while others enable the decoder to perform better error concealment upon detection of errors. Yet another group of techniques are aimed at guaranteeing a basic level of quality and providing a graceful degradation upon the occurrence of transmission errors.

A.1. Robust Entropy Coding

One main cause for the sensitivity of a compressed video stream to transmission errors is that a video coder uses VLC to represent various symbols. Any bit errors or lost bits in the middle of a codeword can not only make this codeword undecodable but also make the following codewords undecodable, even if they are received correctly.

Inserting Resynchronization Markers: One simple and effective approach for enhancing encoder error-resilience is by inserting resynchronization markers periodically. These markers are designed such that they can be easily distinguished from all other codewords and small perturbation of these codewords.

Usually some header information (regarding the spatial and temporal locations or other in-picture predictive information concerning the subsequent bits) is attached immediately after the resynchronization information. This way, the decoder can resume proper decoding upon the detection of a resynchronization marker. Obviously, insertion of resynchronization markers will reduce the coding efficiency: First, the longer and more frequent are such markers, the more bits will be used for them. Second, the use of synchronization markers typically interrupts in-picture prediction mechanisms, such as MV or DC coefficient prediction, which adds even more bits. But longer and frequently inserted markers would also enable the decoder to regain synchronization more quickly, so that a transmission error affects a smaller region in the reconstructed frame. Hence in practical video coding systems, relatively long synchronization codewords are used.

Reversible Variable Length Coding (RVLC): In the above discussion, we have assumed that once an error occurs, the decoder discards all the bits until a resynchronization codeword is identified. With RVLC, the decoder can not only decode bits after a resynchronization codeword, but also decode the bits before the next resynchronization codeword, from the backward direction, as shown in Figure 4. Thus, with RVLC, fewer correctly received bits will be discarded, and the affected area by a transmission error will be reduced. By providing the capability of cross-checking between the output of the forward and backward decoder, at a modest increase in complexity, RVLC can help the decoder to detect errors that are not detectable when non-reversible VLC is used, or provide more information on the position of the errors, and thus decrease the amount of data unnecessarily discarded. RVLC has been adopted in both MPEG-4 and H.263, in conjunction with insertion of synchronization markers.

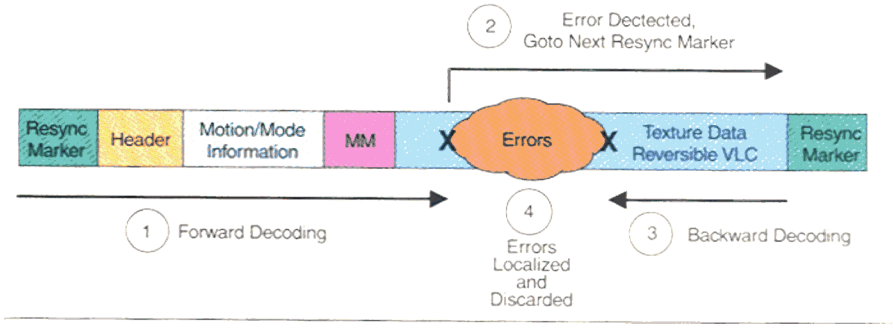

Figure 4 RVLC codewords can be parsed in both the forward and backward direction, making it possible to recover more data from a corrupted data stream. MPEG-4 syntax is assumed in the figure, but the basic priciple holds true also for other RVLC-coded data. From Fig. 9 in [5]

Often, researchers not so familiar with RVLC dismiss the concept due to their belief of the lower coding efficiency of RVLC compared to optimized VLC. Recent research, however, has shown that, at least for coded video, RVLC can be designed with near perfect entropy coding efficiency and nice codeword structure for easy implementation. Intelligently designed RVLC codebooks can also provide a more "robust" and "consistent" performance over a large family of input distributions. This is another advantage over VLC codebooks trained from a specific set of standard sequences whose optimality is questionable when applied to "real" data [17].

In addition to providing cross-checking capability of forward and backward decoded results, RVLC can provide additional error resiliency due to its unique properties. More detailed analysis can be found in [18].

Provisions for Syntax-Based Repairs: Because of the syntax constraint present in compressed video bitstreams, it is possible to recover data from a corrupted bitstream by making the corrected stream conform to the right syntax. Obviously, such techniques are very much dependent on the particular coding scheme. The use of synchronization codes, RVLC, and other sophisticated entropy coding means such as error resilient entropy coding [19] can all make such repair more feasible and more effective.

A.2. Error-Resilient Prediction

Another major cause for the sensitivity of a compressed video to transmission errors is the use of temporal prediction. Once an error occurs so that a reconstructed frame at the decoder differs from that assumed at the encoder, the reference frames used in the decoder from there onwards will differ from those used at the encoder, and consequently all subsequent reconstructed frames will be in error. The use of spatial prediction for the DC coefficients and MVs will also cause error propagation, although it is confined within the same frame. In most video coding standards, such spatial prediction, and therefore error propagation, is further limited to a sub-region (GOB or slice) in a frame.

Insertion of Intra-Blocks or Frames: One way to stop temporal error propagation is by periodically inserting INTRA-coded pictures or MBs. For real-time applications the use of INTRA-frames is typically not possible due to delay constraints. The use of a sufficiently high number of INTRA MBs, however, has turned out to be an efficient and highly scalable tool for error resilience.

When employing INTRA MBs for error resilience purposes, both the number of such MBs and their spatial placement have to be determined. The number of necessary INTRA MBs is obviously dependent

on the quality of the connection. Many practical systems provide out-of-band information about the network quality, or heuristic means to gain such information. Examples include antenna signal strength in wireless environments, or RTCP receiver reports on Internet connections. A good discussion on how to choose the correct number of INTRA MBs is given in [20].

For the spatial placement of I-mode blocks, several schemes have been proposed. Random placement has been shown to be efficient, as well as placement in the areas of highest activity, determined by the average MV magnitude. Hybrid schemes that additionally consider the time of the last INTRA update of a given MB were also considered. None of those schemes outperformed any of the others significantly [21]. The currently best known way for determining both the correct number and placement of INTRA MBs for error resilience purposes is the use of a loss-aware rate distortion optimization scheme [22]. This is further explained in Sec. III.C. Finally, there are closed-loop methods that use a feedback channel to convey information about missing or damaged MB data to trigger INTRA coding at the sender. These schemes are discussed in section II.C below.

Independent Segment Prediction: Another approach to limit the extent of error propagation is to split the data domain into several segments, and perform temporal/spatial prediction only within the same segment. This way, the error in one segment will not affect another segment. One such approach is to include even-indexed frames in one segment and odd-indexed frames into another segment. This way, even frames are only predicted from even frames. This approach is called *video redundancy coding* in [23, 24]. It can also be considered as an approach for accomplishing multiple description coding, to be described in Sec. II.A.4. Another approach is to divide a frame into multiple regions (e.g. a region can be a GOB or slice), and region one, e.g., can only be predicted from region one in the previous frame. This is known as independent segment decoding (ISD) in H.263 (Sec. III.B.5).

A.3. Layered coding with unequal error protection

Layered coding (LC) refers to coding a video into a base layer and one or several enhancement layers. The base layer provides a low but acceptable level of quality, and each additional enhancement layer will incrementally improve the quality. By itself, LC is a way to enable users with different bandwidth capacity or decoding powers to access the same video at different quality levels. Therefore, LC is also called scalable coding. To serve as an ER tool, LC must be paired with unequal error protection (UEP) in the transport system, so that the base layer is protected more strongly, e.g., by assigning a more reliable sub-channel, using stronger FEC codes, or allowing more retransmissions [25,26]. Note that none of the

network and protocol environments discussed in section I.C support UEP at the network level, and this situation is not likely to change in the near future, for both technical and economical reasons. On the other hand, it is feasible to accomplish UEP at the application level, e.g. by packet-based FEC, which will use additional bits.

There are many ways to divide a video signal into two or more layers in the standard block-based hybrid video coder. For example, a video can be temporally down-sampled, and the base layer can include the bitstream for the low frame-rate video, whereas the enhancement layer(s) can include the error between the original video and that up-sampled from the low frame-rate coded video. The same approach can be applied to the spatial resolution, so that the base layer contains a small frame-size video. The base layer can also encode the DCT coefficients of each block with a coarser quantizer, leaving the fine details (the error between the original and the coarsely quantized value) to be specified in the enhancement layer(s). Finally, the base layer may include the header and motion information, leaving the remaining information for the enhancement layer. In the MPEG and H.263 terminology the first three options are known as *temporal, spatial, and SNR scalability*, respectively, and the last one as *data partitioning*.

A.4. Multiple Description Coding

As with LC, multiple description coding (MDC) also codes a source into several sub-streams, known as descriptions, but the decomposition is such that the resulting descriptions are correlated and have similar importance. Any single description should provide a basic level of quality, and more descriptions together will provide improved quality. For each description to provide a certain degree of quality, all the descriptions must share some fundamental information about the source, and thus must be correlated. This correlation is what enables the decoder to estimate a missing description from a received one, and thus provide an acceptable quality level from any description. On the other hand, this correlation is also the source of redundancy in MDC. An advantage of MDC over LC is that it does not require special provisions in the network to provide a reliable sub-channel. For example, in a very lossy network, many retransmissions have to be invoked or a lot of redundancy has to be added in FEC to realize error-free transmission. In this case, it may be more effective to use MDC.

To accomplish their respective goals, LC uses a hierarchical, decorrelating decomposition, whereas MDC uses a non-hierarchical, correlating decomposition. Some approaches that have been proposed for accomplishing such decomposition include overlapping quantization [27], correlated predictors [28], correlating linear transforms [29,30], correlating filter-banks [31], and interleaved spatial-temporal sampling [32,23]. The last approach is known as video redundancy coding in H.263.

B. Decoder Error Concealment

Decoder error concealment refers to the recovery or estimation of lost information due to transmission errors. Given the block-based hybrid coding paradigm, there are three types of information that may need to be estimated in a damaged MB: the texture information, including the pixel or DCT coefficient values for either an original image block or a prediction error block, the motion information, consisting of MV(s) for a MB coded in either P- or B-mode, and finally the coding mode of the MB.

It is well-known that images of natural scenes have predominantly low frequency components, *i.e.* the color values of spatial and temporally adjacent pixels vary smoothly, except in regions with edges. All the techniques that have been developed for recovering texture information make use of the above smoothness property of image/video signals, and essentially they all perform some kind of spatial/temporal interpolation. The MV field, to a lesser extent, also shares the smoothness property, and can also be recovered by using spatial/temporal interpolation. For the coding mode information, the methods developed tend to be more driven by heuristics. In the following, we review some representative techniques for each category. More extensive coverage of methods discussed here as well as other methods can be found in [1, 33, 34].

B.1. Recovery of Texture Information

Motion Compensated Temporal Prediction: A simple and yet very effective approach to recover a damaged MB in the decoder is by copying the corresponding MB in the previously decoded frame, based on the MV for this MB. The recovery performance by this approach is critically dependent on the availability of the MV. When the MV is also missing, it must first be estimated, which is discussed below. To reduce the impact of the error in the estimated MVs, temporal prediction may be combined with spatial interpolation (see below).

Spatial Interpolation: Another simple approach is to interpolate pixels in a damaged block from pixels in adjacent correctly received blocks. Usually, because all blocks or MBs in the same row are put into the same packet, the only available neighboring blocks are those above and below. Because most pixels in these blocks are too far away from the missing samples, usually only the boundary pixels in neighboring blocks are used for interpolation [35]. Instead of interpolating individual pixels, a simpler approach is to estimate the DC coefficient (i.e. the mean value) of a damaged block and replace the damaged block by a constant equal to the estimated DC value. The DC value can be estimated by averaging the DC values of

surrounding blocks, or the ? -trimmed mean of pixels in the neighboring blocks [36]. One way to facilitate such spatial interpolation is by an interleaved packetization mechanism so that the loss of one packet will damage only every other blocks or MBs.

Spatial and Temporal Interpolation by Maximizing the Smoothness of Resulting Video: A problem with spatial interpolation approaches is how to determine appropriate interpolation filter. Another shortcoming is that it ignores received DCT coefficients, if any. These problems are resolved in the approach of [37] by requiring the recovered pixels in a damaged block to be smoothly connected with its neighboring pixels both spatially in the same frame and temporally in the previous/following frames. If some but not all DCT coefficients are received for this block, then the estimation should be such that the recovered block be as smooth as possible, subject to the constraint that the DCT on the recovered block would produce the same values for the received coefficients. The above objectives can be formulated as an unconstrained optimization problem, and the solutions under different loss patterns correspond to different interpolation filters in the spatial, temporal, and frequency domains.

Spatial Interpolation Using Projection onto Convex Sets (POCS) Technique: Another way of accomplishing spatial interpolation is by using the POCS method [38, 39]. The general idea behind POCS-based estimation methods is to formulate each constraint about the unknowns as a convex set. The optimal solution is the intersection of all the convex sets, which can be obtained by recursively projecting a previous solution onto individual convex sets. When applying POCS for recovering an image block, the spatial smoothness criterion is formulated in the frequency domain, by requiring the discrete Fourier transform (DFT) of the recovered block to have energy only in several low frequency coefficients. If the damaged block is believed to contain an edge in a particular direction, then one can require the DFT coefficients to be distributed along a narrow stripe orthogonal to the edge direction, i.e., low-pass along the edge direction, and all-pass in the orthogonal direction. The requirement on the range of each DFT coefficient magnitude can also be converted into a convex set, so is the constraint imposed by any received DCT coefficient. Because the solution can only be obtained through an iterative procedure, this approach may not be suitable for real-time applications.

B.2. Recovery of Coding Modes and Motion Vectors

As already indicated, some of the above algorithms are contingent upon the knowledge of the coding mode and MVs for P- or B-mode MBs. To facilitate decoder error concealment, the encoder may perform data partition, to pack the mode and MV information in a separate partition and transmit them with more

error protection. This for example is an ER mode in both H.263 (Sec. III.B.6) and MPEG4 (Sec. IV.B). Still, the mode and MV information can be damaged. One way to estimate the coding mode for a damaged MB is by collecting the statistics of the coding mode pattern of adjacent MBs, and find a most likely mode given the modes of surrounding MBs [40]. A simple and conservative approach is to assume that the MB is coded in the INTRA-mode, and use only spatial interpolation for recovering the underlying blocks [37].

For estimating lost MVs, there are several simple operations [41]: (a) assuming the lost MVs to be zeros, which works well for video sequences with relatively small motion; (b) using the MVs of the corresponding block in the previous frame; (c) using the average of the MVs from spatially adjacent blocks; (d) using the median of MVs from the spatially adjacent blocks; (e) re-estimating the MVs [36]. Typically, when a MB is damaged, its horizontally adjacent MBs are also damaged, and hence the average or mean is taken over the MVs above and below. It has been found that the last two methods produce the best reconstruction results [36,20,42]. Instead of estimating one MV for a damaged MB, one can use different MVs for different pixel regions in the MB for a better result. Readers are referred to [1,33] for some more sophisticated approaches.

C. Encoder and Decoder Interactive Error Control

In the techniques presented thus far, the encoder and decoder operate independently as far as combating transmission errors is concerned. When a feedback channel can be set-up from the decoder to the encoder, the decoder can inform the encoder about which part of the transmitted information is corrupted by errors, and the encoder can adjust its operation correspondingly to suppress or even eliminate the effect of such errors. If the underlying network protocol supports ARQ, then a simple approach is to retransmit the lost packets. This however will incur processing delays that may be unacceptable for certain real-time interactive applications. For such applications, it is often acceptable to have errors as long as they do not last too long. Therefore, even if one cannot afford to correct the errors whenever they occur, it is important to limit the propagation scope of such errors. In Sec. II.A, we described how to use periodic INTRA blocks/frames or independent segment prediction in the encoder to limit error propagation. These approaches however, generally lead to significant reduction in coding efficiency. Here, we describe some techniques that adjust the encoder operations based on the feedback information from the decoder. These approaches can reduce the coding gain loss, at increased complexity. A more extensive review of techniques in this category can be found in [43].

C.1. Reference Picture Selection (RPS) based on Feedback Information

One way of taking advantage of an available feedback channel is to employ RPS. If the encoder learns through a feedback channel about damaged parts of a previously coded frame, it can decide to code the next P-frame not relative to the most recent, but to an older reference picture, which is known to be available in the decoder. This requires that the encoder and decoder both store multiple past decoded frames. Information about the reference picture to be used is conveyed in the bitstream. Compared to coding the current picture as an I-frame, the penalty for using the older reference picture is significantly lower, if the reference picture is not too far away. Note that using RPS does not necessarily mean extra delay in the encoder. The encoder does not have to wait for the arrival of the feedback information about the previous frame to code a current frame. Rather, it can choose to use as reference a frame before the damaged frame whenever it receives the feedback information. For example, if the information about the damage of frame *n* does not arrive at the encoder until the time to code frame *n+d*, the decoded frames between *n+1* to *n+d-1* would all have errors, because the decoder uses different reference frames than the encoder for these frames. By selecting frame *n-1* as the reference frame to code frame *n+d*, error propagation will be stopped from frame *n+d* onwards. Of course, the longer it takes to generate and deliver the feedback information, the greater will be the loss in the coding efficiency.

C.2. Error Tracking based on Feedback Information

Instead of using an earlier, undamaged frame as the reference frame, the encoder can track how the damaged areas in frame *n* would have affected decoded blocks in frames *n+1* to *n+d-1*, and then perform one of the following. The encoder can (a) code the blocks in frame *n+d* that would have used for prediction damaged pixels in frame *n+d-1* using the INTRA-mode; (b) avoid using the affected area in frame *n+d-1* for prediction in coding frame *n+d*; and (c) perform the same type of error concealment as the decoder for frames *n+1* to *n+d-1*, so that the encoder's reference picture matches with that at the decoder, when coding frame *n+d*. The first two approaches only require the encoder to track the locations of damaged pixels or blocks, whereas the last approach requires the duplication of the decoder operation for frames *n+1* to *n+d-1*, which is more complicated. In either approach, the decoder will recover from errors completely at frame *n+d*. Option (a) is the simplest to implement, but may suffer higher coding gain loss than the other two options. More information on error tracking, correction and fast algorithms can be found in [44, 45, 43, 1]. Experimental results have shown that option (a) is less efficient than the RPS approach [46].

III. Error Resilience Tools in ITU-T H.263

A. Introduction

As is the case for all current video coding standards, the ITU-T Recommendation H.263 entitled "Video coding for low bitrates" [47] defines the bitstream syntax and the decoder operation on that syntax. Neither is the encoder operation defined, nor the decoder's reaction on corrupted bitstreams. During the standardization process of H.263, a test model description emerged, which covers these two aspects. The test model's purpose is both to provide ideas and guidelines to implementers, and to describe common grounds of video coding based on the H.263 syntax that is used for future standardization work.

This section is based on both the forthcoming third version of the ITU-T Recommendation H.263, known commonly as H.263++, and on the most recent test model version 11 known as TMN11 [48]. H.263++ was technically frozen in February 2000, and it is expected to become an international standard by October 2000. The test model was finalized at the same time.

H.263 follows the general ideas of block-based hybrid coding described in Sec. I.B. Beyond the baseline syntax, H.263 offers a variety of optional coding modes that adjust various tradeoffs. Some of these modes are intended to improve error resilience by adding redundancy to the bitstream, and they are discussed in more detail below. The other optional modes typically allow adjusting the tradeoff between computational complexity (at the encoder, the decoder, or both) and the compression effectiveness. None of those modes have more than a marginal impact on error resilience beyond the general observation that the better the compression of a signal, the more bits can be spent for the transport coder to improve the reproduced signal quality at the receiver. Therefore, those coding-efficiency oriented modes are not discussed here in detail. An excellent overview of the version 2 optional modes is available in [49]. The only coding efficiency optional mode of version 3 was introduced in detail in [50].

We introduce first the ER tools available in version 3 of H.263. Then, we discuss the application of these tools at the encoder for Internet-based packet lossy networks and for highly bit error prone mobile networks. We will introduce the RTP packetization scheme used for H.263 transport over the Internet in real-time environments, as important tools such as header repetition are implemented there and not in the video bitstream syntax. Typical decoder operations for both networking scenarios are also presented, including mechanisms for error concealment and syntax repair, where appropriate.

B. Error resilience tools

H.263 version 3 contains, organized in eight annexes, five error resilience tools: block-based forward error correction (FEC), flexible synchronization points (Slices), independent segment decoding (ISD), reference picture selection (RPS), and data partitioning syntax (DP). As discussed in Sec. II.A.3, the temporal, spatial, and SNR scalability modes can also be used to support error resilient applications. An appropriate combination of these tools along with means available in the baseline syntax, such as INTRA MB refresh, is typically chosen adaptively by the application according to the network characteristics and conditions. See section III.C below for such combinations. Additional remarks on the effectiveness of many of the tools can be found in [24]. In the following each of the tools is discussed in the order they appear in the ITU-T recommendation.

B.1. BCH forward error correction (FEC) (Annex H)

This tool allows including, for blocks of 492 coded video bits, a 19 bit BCH (511, 492) FEC parity information in the bitstream. Together with a single additional bit to allow for resynchronization to the resulting 512 bit block structure, Annex H introduces an overhead of roughly 4% of the bitrate. The BCH (511, 492) FEC code is able to correct single, and to reliably detect double bit errors in a 512 bit block.

Annex H is a leftover from H.261, where it was deemed to be necessary to cope with the bit errors of ISDN. As this network practically does not have bit errors (BER of 10E-8 is guarantied, and, typically BERs of 10E-10 are achieved), it was rarely used in this context. For highly bit error prone mobile channels with their bursty error characteristics, Annex H is not efficient since an error burst longer than 2 bits is neither correctable, nor will reliably be detected. Furthermore, the fixed structure of 492 video bit blocks does not allow to precisely align the block boundaries to synchronization markers. For these reasons, Annex H is not used for the two networking scenarios discussed.

B.2. Flexible synchronization marker insertion, Slice Structured Mode (Annex K)

In its Annex K, H.263 introduces a slice concept that goes well beyond what is known as slices from the MPEG world. These slices, when used, replace the GOB concept in H.261 and baseline H.263. In particular, slice headers that serve as synchronization markers and interrupt in-picture prediction, can either be inserted in scan order to achieve picture structures similar to those of MPEG-2, or slices can be used to define rectangular areas of the picture, aligned to MB boundaries (cf. Figure 5). In conjunction

with Independent Segment Decoding, as discussed below, such Rectangular Slices are helpful for certain continuous presence multipoint applications.

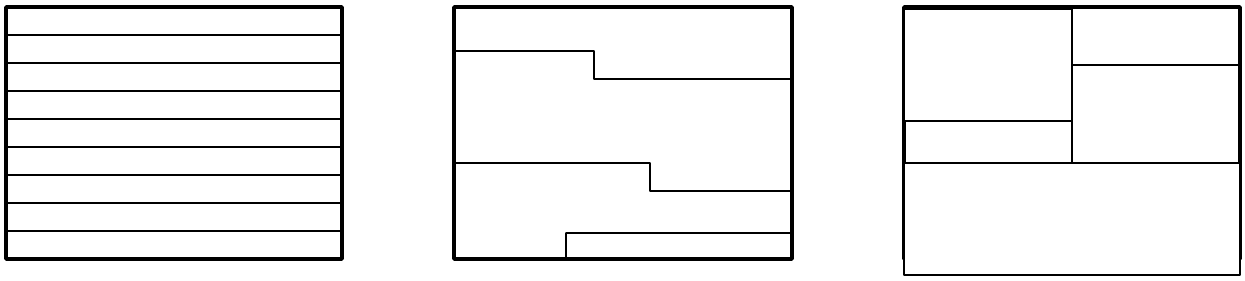

Figure 5: H.263 Annex K Slices in a QCIF picture: GOBs (left) consists of one line of MBs. Scan order slices (center) may contain between one MBs and all MBs of a picture. Rectangular slices (right) can be placed at MB boundaries in any rectagular shape. In each case, all MBs of a picture have to be assigned to exactly one GOB/Slice.

Furthermore, data from both forms of slices is allowed to appear either in scan-order only, or in any arbitrary order. Limiting the appearance of slices to the scan order facilitates error detection at the decoder. Slice interleaving schemes, like the one discussed later in conjunction with Internet packetization, however, cannot be easily used. Out-of-order slices allow for more flexibility in the bitstream generation and de-packetization, but are more difficult to implement and might also induce a somewhat higher delay, as missing data cannot be identified before the next picture start is reached – although such a task is typically performed by the de-packetization at the transport layer without additional delay.

A simple example on how can scan order slices improve error resilience is visualized in Figure 6. Using GOB headers for every other GOB inserts 4 resynchronization markers in the coded bitstream, but at positions determined by spatial locations in the source picture and not by the content. Inserting slice headers as synchronization markers, however, leads to an equal size of bits in each slice. This, by itself, improves error resilience as it equalizes the probability that a slice be hit by a transmission error. Error resilience is furthermore improved, because a higher number of synchronization markers appear in the active region of the picture, which can, therefore, be reconstructed with a higher probability.

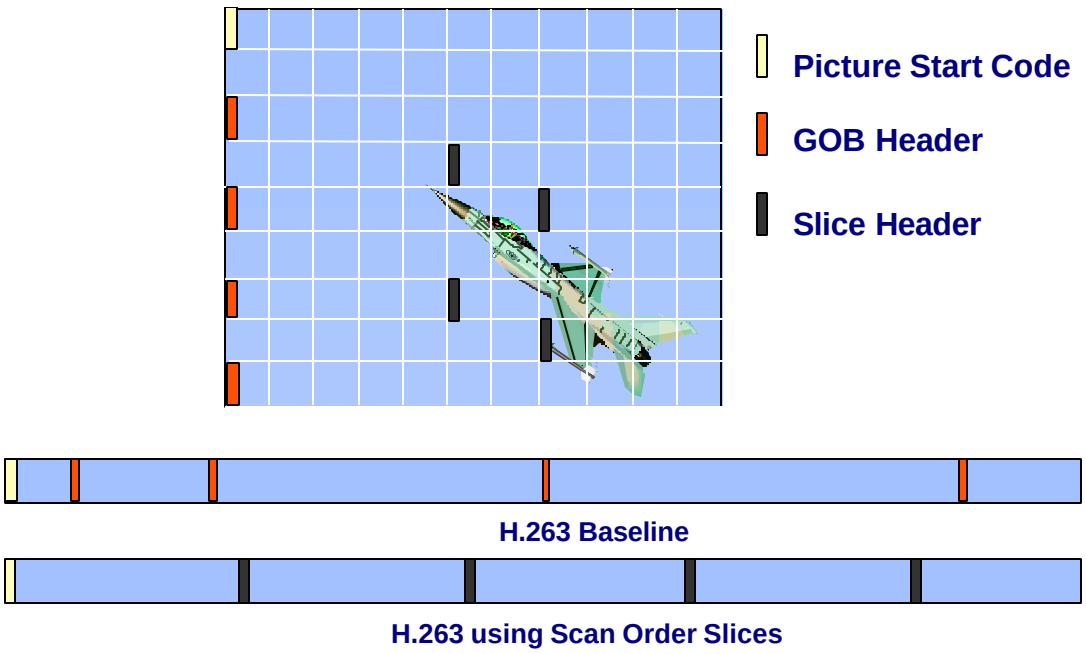

Figure 6: Scan order slices as a means to improve error resilience.

B.3. Reference Picture Selection (RPS -Annex N and Annex U)

RPS allows importing picture data from more than one reference picture, which are typically held in a first-in, first-out queue. The basic concept of RPS can be most easily described as the use of three-dimensional MVs. Predictive information can not only be imported, after spatial movement, from the previous picture, but from more than one older pictures that have to be kept in storage at the decoder (a prediction model that expresses the pixel value at the current pixel as the linear combination of pixels in a number of previous frames was developed in [[51]]). RPS can be used on whole pictures, picture segments (slices or GOBs), or on individual MBs. The former two mechanisms are defined in Annex N and were included as an ER tool only, whereas the latter, defined in Annex U, is a version 3 extension and was designed with both error resilience and coding efficiency in mind. The main difference between these concepts is the signaling in the bitstream. In case of RPS operation on whole pictures or picture segments, the to-be-used reference picture information needs to be transmitted only once per picture or picture segment. When using MB-based RPS, every coded MB has to contain this information, thereby yielding three-dimensional MVs with the reference picture time as the third dimension. In all cases, both the encoder and the decoder have to maintain data structures with more than one reference picture.

For error resilience purposes, RPS can be used with or without feedback information. Without feedback information, a temporal MDC technique known as video redundancy coding is employed. This technique is, however, much less efficient than any feedback-based mechanisms [23,24].

If a feedback channel is available, a decoder can inform the encoder of a missing or damaged picture. The encoder can react to such a message by using an older reference picture that is known to be correctly received at the decoder's site. As already mentioned, the induced penalty for the less accurate prediction depends on the age of the employed reference picture. Reference [43] contains information about delay and overhead incurred by using RPS. The original concept was introduced as the NEWPRED method in [52].

When using a feedback based mechanism, immediately a need arises for the transport of the feedback information. From an architectural point of view, this functionality is outside the scope of video source coding. For reasons that lie in the history of the development of H.263 version 2 and the original version of H.324, Annex N of H.263 includes syntax for feedback channel information in the video stream. This limits the use of that flavor of feedback channel messaging to only point-to-point scenarios in which both terminals use H.263 for video coding. Most systems employing feedback-based RPS rely, however, on the transmission of feedback channel information outside of the video stream, by higher protocol layer means. This has not only architectural advantages, but also allows using a different service quality for feedback channel messages thereby avoiding problems resulting from lost feedback channel messages.

B.4. Scalability (Annex O)

Temporal, Spatial, and SNR scalability (see Sec. II.A.3) can be used for error resilience purposes as well, if multiple paths with different service characteristics are available between the source and the destination. Annex O implements the same scalability means that were already discussed in section II.A.3, and is therefore not further discussed here.

B.5. Independent Segment Decoding (ISD - Annex R)

In certain high bitrate, packet lossy environments, ISD in conjunction with rectangular slices is known to improve error resilience [24]. ISD forces encoder and decoder to treat segment (slice or GOB) boundaries like picture boundaries, thereby preventing the import of corrupted data outside the segment due to motion compensation. The overhead of ISD is roughly reversely proportional to the picture size, and is impractically high for picture sizes smaller than CIF.

B.6. Data Partitioning and RVLC (Annex V)

Because of the effectiveness of data partitioning and RVLC in error resilience, they are adopted as an ER mode in Annex V of H.263 version 3. The framework proposed in Annex V bears much similarity to the well-understood H.263 baseline, with the following modifications:

1) MB header, MV, and DCT information are no longer interleaved on a MB by MB basis, rather, they are grouped into separate partitions separated with specially designed markers [53].

2) Header and motion information is coded with RVLC. The RVLC table for motion information was already introduced in the H.263 Version 2 as part of Annex D (Unrestricted Motion Vector Mode [54]). The RVLC table used for header information is a symmetric code table, which has very similar code length distribution as the non-reversible VLC table used in H.263 baseline. Because DCT information usually has a much smaller impact on image quality as compared to header and motion information, DCT information is still coded with the table used in the baseline.

3) To improve the coding efficiency, MVs for MBs in a packet are predictively coded in Annex V, as in the case for baseline H.263. However, the predictor used is no longer drawn from the MVs of 3 neighboring MBs, but rather, a new single thread MV prediction scheme [55] is used. This scheme allows backward independent MV (not just MV prediction residual) decoding in both the forward and backward directions at little loss of efficiency.

4) Annex V uses a pseudo-fixed-length packetization scheme so as to help the decoder to regain correct resynchronization.

B.7. Header repetition (Annex W)

Annex W of H.263 contains several mechanisms to convey supplementary enhancement information additional to those of Annex L. Annex L was introduced in version 2 of H.263 and includes a set of simple signals for mechanisms such as picture freezing or color keying – it does not include any error resilience related mechanisms and is therefore not further discussed. A very simple signaling mechanism allows extending the picture header by such enhancements. For error resilience purposes, a redundant representation of either the current or the previous picture header can be included. When using a redundant implementation of the current picture header, then the problem arises that, syntactically, both headers reside in the same data structure, namely the picture header itself. It is therefore difficult, and sometimes impossible, to identify the redundant information, if the original information is damaged.

Using the previous header such a problem does not arise, but at the expense of additional delay, as the picture header of the next picture has to be available before decoding of the previous picture can start.

H.263 contains no syntax element that allows including redundant picture header information at the slice level, as it is available in MPEG-4's HEC. This serious drawback, necessary due to the syntax structure, is somewhat compensated by accompanying protocols such as RFC2429 [56], where such redundant headers are possible.

C. Encoder mechanisms used on IP networks

On IP networks, interactive video capable applications typically employ RTP [9] as the transport protocol, which offers real-time, packet lossy, bit error free, and unreliable service. On top of RTP, RFC2429 is the payload header specification for H.263. Source coding mechanisms of H.263, and packetization tools of RFC2429 work together to achieve good performance. If longer latencies are acceptable, either guarantied transmission protocols such as TCP can be employed or, in the case of multicast and broadcast environments, often a combination of unreliable transmission with a reliable ARQ feedback and forward channel is used. The latter is, for example, common in tools like the RealPlayer ™. As we focus here on low latency environments such algorithms are not discussed. In the following, we briefly describe an algorithm used in the test model [48,22]. It does not rely on feedback mechanisms and, therefore, allows for multicast and broadcast applications, like the ones common in Multicast Backbone (MBONE) environments. Whenever the application allows the use of feedback channels, RPS, ARQ, or other mechanisms currently not employed in the test model will likely yield better results then this algorithm.

At the source coding level, an adaptive INTRA MB refresh algorithm is used, which employs a loss-aware rate-distortion optimization scheme to select the MBs to be INTRA coded. This algorithm takes as an input a mid-term prediction of the packet loss rate. Each MB is coded INTRA, INTER, and SKIP, and the resulting rate-distortion tradeoff assuming undamaged reconstruction of the MB is measured. Then, for the same coding modes, the same rate-distortion tradeoff is calculated but under the assumption that the coded MB got lost during transmission. For each coding mode (INTRA, INTER, SKIP), both values are weighted according to the packet loss rate, to decide on the coding mode. A detailed description of the algorithm along with simulation results can be found in [22]. Additionally, the picture is divided into GOB-shaped slices, which can be decoded out-of-order. The resulting slices are packetized in an interleaved manner, collecting all MBs residing in even numbered rows into one and all MBs in odd numbered rows into another packet. The "odd" packet contains the original picture header. For the

"even" packet, a redundant representation of the picture header is included, which allows the decoding of the picture even if the first packet containing the original picture header is lost during transmission. The interleaving scheme allows for decent error concealment using motion compensated temporal interpolation, as discussed in section III.D below. It can be extended to more than two packets per picture if desirable, which is typically only the case for video bitrates above 300 kbit/s. Figure 7 displays the packetization of a QCIF-sized picture into two RTP packets.

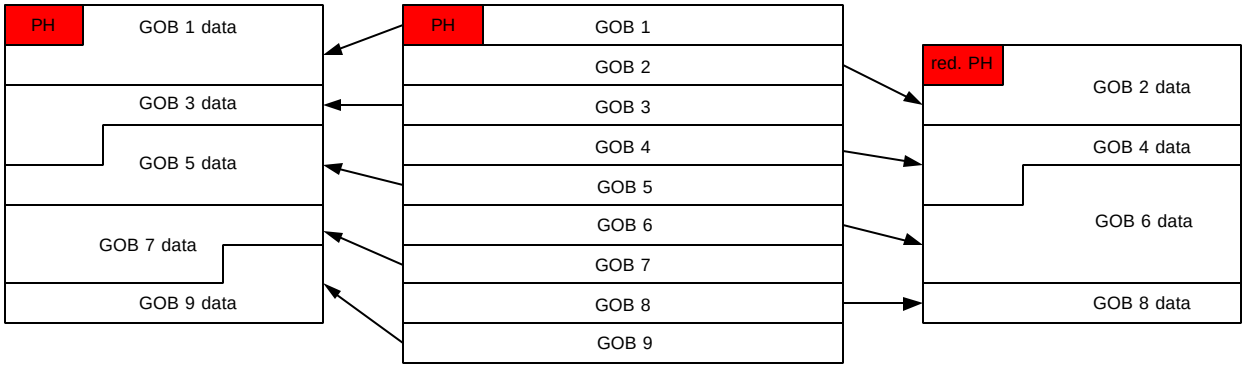

Figure 7: A QCIF picture is coded into 2 packets. The packet not containing the first GOB/Slice (that includes the Picture Header) carries a redundant copy of the Picture Header in the RFC2429 RTP payload header.

D. Decoder operations for IP-based video transmission

On IP-based networks, the transport protocol (here UDP) typically guaranties bit error free transmission by checksum mechanisms and dropping of any corrupted packets. Therefore, the only error type that has to be discussed here is the packet loss. Furthermore, payload/overhead considerations lead to fairly large packets around the Internet MTU size of 1500 bytes.

Due to the discussed network characteristics, syntax-based repair mechanisms can only ensure a standard compliant bitstream, but not reconstruct missing data. Most decoders for IP-based video therefore do not perform syntax-based repair at all, but use the syntax violations to detect missing MB data to apply error concealment.

Without feedback channels, and when using the corresponding encoder design as discussed above, then the decoder design is fairly simple. If all packets of a coded picture are received, which can be determined by checking RTP header information, then the concatenated payload of those packets can be directly decoded, as it forms a valid H.263 bitstream. If only the first packet of a given picture is received correctly, then the correct picture header for this picture is available, which is part of the first slice of this picture. In H.263, all MBs have to be coded in each picture, even if no texture or motion information is conveyed. Macroblocks missing in the picture are a syntax violation, and a decoder can react to this violation by employing content-based error concealment techniques, such as predicting presumably missing MVs from neighboring MVs. The test model suggests to use the MV of the MB spatially above of the missing MB. Research has shown that no significant improvement in picture quality can be achieved when using mean or median values of more than one MV. The interleaving scheme discussed in Sec. III.C ensures that such a prediction is often possible, as neighboring lines of MBs are coded in different packets.

If the first packet of a picture is lost, redundant representations of the picture header in other packet(s) can be used to reconstruct the picture header and thus still decode parts of the picture. Even if no such redundant copy is available, a decoding attempt can still be made by a highly error tolerant decoder, because, typically, little changes can be observed in the picture header from picture to picture. The only field that always changes is the Temporal Reference, and this field can be reconstructed out of the RTP timestamp, a mandatory part of the RTP packet header.

Any missing packets lead, of course, to an asynchrony between the encoder's reference picture and the decoder's reference picture. As discussed earlier, such inconsistencies have negative impact on the reproduced picture quality. To cope with this asynchrony, either RPS, INTRA MB refresh, or a combination of both can be used, as discussed in Sec. III.C.

In case of an available feedback channel, decoder operation is very similar to the above description, except that the decoder has to generate feedback information. Both positive (Ack) and negative acknowledgments (Nck) are supported in H.263. For optimal performance the choice between using Ack, Nack, or both signals should be made according to the transmission quality for the video and for the reverse feedback channels [52, 24, 43]. Obviously, the smaller is the spatial region on which feedback information is generated, the higher will be the traffic on the feedback channel. Therefore, Ack and Nack type of feedback is typically generated for whole pictures, whereas for picture segments or individual MBs, Nack-only type is more appropriate.

As an encoder reacts appropriately upon receiving a feedback message, e.g., by using an older reference picture, the encoder loop and decoder loop regain synchronicity as soon as the RPS-coded picture is received. Therefore, INTRA MB refresh rates can be kept to a minimum, thereby improving coding efficiency and resulting picture quality.

E. Encoder and decoder mechanisms for highly bit error prone networks

On highly bit error prone networks, TMN11 suggests the use of essentially the same mechanisms as for the Internet. Due to the low overhead characteristics of the transport protocol used for ITU-T-compliant mobile terminals (H.223), the packet sizes can be significantly smaller. TMN11 suggests to packetize one coded GOB into a single packet. In the future we expect the use of Annex V, as data partitioning and RVLC are known to be efficient in this scenario and were for this reason added to the standard. No test model enhancements are available at the time of publication.

As already discussed above, highly bit error prone channels involve typically two types of errors in the video payload: bit errors and losses of a large group of bits due to errors in higher level protocols. The latter type of error is called packet loss here. Video codecs have to cope with both types of errors simultaneously, or can often request the transport hierarchy to discard any packets containing bit errors. In the latter case many bits containing useful information are generally discarded, but decoder implementation is much simpler and follows the rules discussed above.

Individual bit errors or short error bursts can be subject to syntax repair mechanisms. Before an erroneous bitstream can be repaired, it is necessary to identify the exact position of the error, which, in turn, can be done either completely based on syntax, or using a mixture of syntax- and semantic-based mechanisms. The easiest way to repair a broken bitstream is by switching individual bits and trying to reconstruct that repaired bitstream, until no syntax violation occurs any more. More sophisticated repair mechanisms were also reported.

The repaired bitstream is either completely free of bit errors, or contains error bursts that are too long so that they cannot be repaired. In such cases, decoders can attempt to decode as many bits as possible, at the risk of reconstructing pictures from erroneously transmitted bits, or they can discard all bits between two sync-points where a bit error is known to be present – which effectively results in a packet loss. Both mechanisms have advantages and disadvantages, and typically result in a quite similar system behavior.

For the packet losses, mechanisms very similar to those discussed in Secs. III.C and III.D can be employed, including INTRA MB refresh, RD-based mode decision processes, RPS, and header repetition and use of redundant header information. Upon detection of a missing or intentionally dropped packet, error concealment techniques discussed in Sec. III.D can be employed.

IV. Error Resilience Tools in MPEG-4

The latest video coding standard from ISO/IEC, the MPEG-4 standard, was aimed at audio-visual coding in multimedia applications, allowing for interactivity, high compression and/or universal accessibility and portability of audio and video content. The targeted bitrate is between 5-64 Kbps for mobile applications and upto 2 Mbps for TV applications. The MPEG-4 standard utilizes the concept of object-based and layered encoding extensively, as well as tools that can deal with both natural and synthetic objects. To reduce overhead and maintain compatibility to other video coding standards, MPEG-4 also contains a short header mode that is essentially the same as the H.263 baseline syntax when only a single video object is present. The MPEG-4 standard also incorporates explicitly an error resilient mode of operation. It incorporates many error isolation, synchronization, data recovery and resilient entropy coding tools. In this section, we focus on ER tools introduced for coding the motion and texture information. Section V will describe options introduced for shape coding.

A. MPEG-4 Resynchronization Tools

Resynchronization tools, as the name implies, attempt to enable resynchronization between the decoder and the bitstream after errors have occurred in the bitstream. This is especially helpful in the case of bursty errors as it provides the decoder with the capability of "fresh start".

There are several different approaches to resynchronization in MPEG-4. The video packet approach adopted by MPEG-4 is very similar in principle to the adaptive slice of MPEG-2 and Slice Structured Mode of H.263 (Sec. III.B.2). It is aimed at providing periodic resynchronization throughout the bitstream. Therefore, when the video packet approach to resynchronization is used, the length of the video packets are no longer based on the number of MBs as in the case of the non-error resilient mode of MPEG-4 or baseline H.263, but instead on the number of bits contained in that packet. If the number of bits contained in the current video packet exceeds a predetermined threshold, then a new video packet is created at the start of the next MB.

The resynchronization marker placed at the start of a new video packet is distinguishable from all possible VLC codewords. Header information is also provided at the start of a video packet. This header contains the information necessary to restart the decoding process, including the address of the first MB contained in this packet and the quantization parameter (QP) necessary to decode that first MB. The MB number provides the necessary spatial resynchronization while the QP allows the differential decoding process to be resynchronized. Following the QP is the Header Extension Code (HEC). As the name implies, the HEC is a single bit to indicate whether additional video object plane (VOP) level information will be available in this header. If the HEC is equal to one then the following additional information (which is constant for each VOP and transmitted with the VOP header) is available in this packet header: timing information, temporal reference, VOP prediction type, and some other information. The header extension feature enables the decoder to correctly utilize data contained in the current packet without reference to the packet containing the VOP header, it can also help error detection because it offers cross-checking capability since all packets in the same VOP share the same QP, time stamp, etc.

In addition to the video packet approach to resynchronization, a method called fixed interval synchronization has also been adopted by MPEG-4. This method requires that the start of a video packet appear only at allowable, fixed interval locations in the bitstream. This helps to avoid the problems associated with start code emulation. Although errors can cause emulation of a VOP start code, this emulation will only be problematic in the unlikely event that it occurs at a location permitting VOP start codes.

B. MPEG-4 Data Partitioning

In the absence of any other ER tools, the data between the synchronization point prior to the error and the first point where synchronization is re-established is discarded when errors are detected in the decoding of "real" data. If the synchronization approach is effective in determining the amount of data discarded by the decoder, then the ability of other types of tools that recover data and/or conceal the effects of errors can be greatly enhanced.

To achieve better error isolation in the video packet and fixed interval synchronization approaches, MPEG-4 introduced data partitioning. When the data partitioning syntax is used, video bitstream between two consecutive resynchronization markers is divided into finer logical units. Each logical unit contains one type of information (e.g. DCT coefficients) for *all* the MBs in the *whole* packet (when present, shape data is also partitioned). This is in contrast to the non-data-partitioned syntax, in which header, motion

and texture information are interleaved on a MB by MB basis. For the decoder to locate each logical unit, secondary markers are placed between logical units. Unlike the resynchronization marker, which needs to be free of emulation from header, motion and DCT data, these secondary markers need only to be free from emulation by data in the logical unit that immediately precedes them. For example, the marker between motion and DCT data needs only to be free from emulation by motion data, but it can be emulated by DCT data. When the decoder detects an error in a packet using the data partitioning syntax, it can then search for the next secondary marker in the packet, and start decoding the next logical unit within the same packet. Because the decoder only needs to discard the rest of the logical unit, instead of the rest of the packet, more data can be salvaged and utilized. Without data partitioning, the decoder would need to compensate for the loss of header *and* motion *and* DCT data for *all* MBs from the one in which the error is detected. When data partitioning is used, each correctly decoded logical unit contains one type of information for *all* MBs in the packet, the task of error concealment is thus made much easier. Figure 8 is an illustration of the syntactic structure of the MPEG-4 error resilient mode with data partitioning. Note that the texture (DCT) partition in the structure may be coded with RVLC.

Resync Marker	MB Index	Quant Info	HEC and Header Repetition (If Present)	Motion Vector	Motion Marker	Texture (DCT)	Rysnc. Marker	…

Figure 8: Syntactic Structure of Packets in MPEG-4 Error Resilient Mode with Data Partitioning.

C. NEWPRED mode

The "NEWPRED" mode is a new ER tool introduced to the MPEG-4 Version 2 (officially known as MPEG-4 Visual Amendment 1, Visual Extensions) [57]. It is very similar in principle to the H.263 RPS Mode (Annex N) and the Slice Structured Mode (Annex K). When the NEWPRED mode is turned on in MPEG-4, the reference used for INTER prediction by the encoder will be updated adaptively according to feedback from the decoder via feedback messages. These upstream messages indicate which NEWPRED (ND) segments (which can either be an entire frame, or in MPEG-4 language, a VOP, or the content of a packet, typically one slice) have been successfully decoded, and which ND segments have not. Based on the feedback information the encoder will either use the most recent ND segment, or a spatially corresponding but older ND segment for prediction. In the latter case the coding efficiency is reduced, as longer MVs and additional texture information will typically have to be used.

D. Data Recovery in Conjunction with RVLC and Data Partitioning

After synchronization has been re-established, data recovery tools attempt to recover data that would otherwise be lost. The MPEG-4 error resilient operation mode utilizes RVLC for better recovery of DCT data. The usage of RVLC is signaled at the Video Object Layer (VOL). When RVLC is used for DCT data, the bitstream is decoded in the forward direction first. If no errors are detected, the bitstream is assumed to be valid. If an error is detected however, two-way decoding is applied, and the portion of the packet between the first MB in which an error was detected in both the forward and backward directions should not be used.

In RVLC decoding, errors are detected if either an illegal RVLC codeword is found or more than 64 DCT coefficients are decoded in a block. Here "an illegal RVLC" is detected upon receiving a bit pattern not listed in the code table, or from syntactical information (i.e. an incorrect number of stuffing bits for byte alignment, fixed length coded EVENTs already included in RVLC table, etc.)

Syntax-Base Repairs: Because RVLC provides more constraints and check points, it is very suitable for syntax based repair, as introduced in [58, 59]. It is a widely held misconception that when a VLC code table is complete (i.e. when the Kraft's inequality is satisfied as an equality), any binary sequence becomes a legal concatenation of codewords. This conception is true only when the packet (or data segment) is of infinite length, which is never the case in a real system. In addition, a standard (MPEG or H.26x) compliant bitstream is never an i.i.d. white binary sequence. The distribution of 1's and 0's in the bitstream is often (albeit slightly) location dependent. There is also a syntax constraint on the concatenation of codewords. Therefore, for any given packet, there is only a relatively small number of valid, legal combinations. Thus, if a decoder is able to use this information intelligently and efficiently, it is possible to achieve much superior quality than a "traditional" video decoder when the bitstream is transmitted over a noisy channel [58,59]. The MPEG-4 data partitioned syntax and RVLC also help to improve the benefit of syntax-based repair, as they allow more "check points" and syntax constraints, which makes the total number of valid packets even smaller as compared to the number of all possible binary strings of the same length. Figure 9 is an illustration of the improvements provided by a syntax-base repair algorithm similar to that in [58] for a MPEG-4 bitstream corrupted with the ITU standard W-CDMA error pattern with an average BER of 10^{-3}.

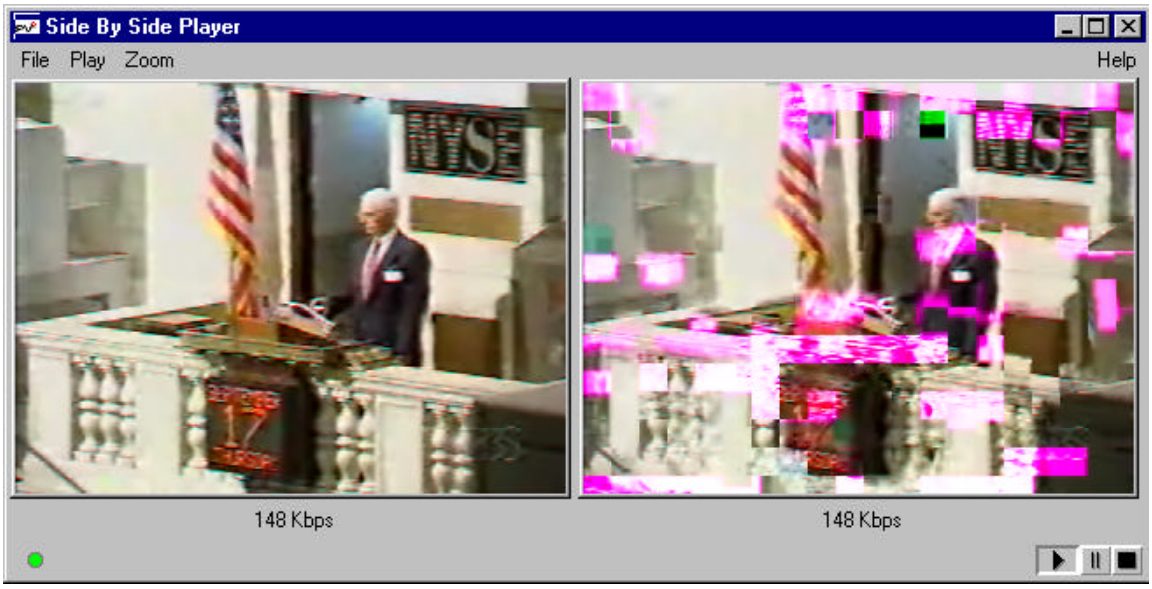

Figure 9: A Comparison of Reconstructed Video Quality with and without Syntax Based Repair (Left: With syntax based repair, Right: Without syntax based repair. Identical bitstreams and error detection and concealment were used in both cases).

V. Error Resilience for Shape Coding in MPEG-4

A. Overview of the MPEG-4 Shape Coding Methods

One of the dramatic differences of MPEG-4 from all previous video coding standards is the coding of arbitrarily shaped video objects (VOs) [60]. A frame of a VO is called a video object plane (VOP). Following an object-based approach, MPEG-4 visual transmits texture, motion, and shape information of one VO within one bitstream. The bitstreams of several VOs and accompanying composition information can be multiplexed such that the decoder receives all the information to decode the VOs and arrange them into a video scene, as illustrated in Figure 10. This results in interactivity and flexibility for standardized video and multimedia applications not shared by any of the previous coding standards.

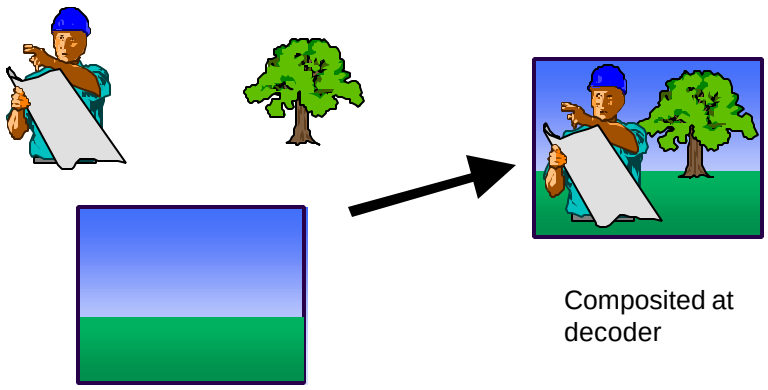

Figure 10: Frame Decomposition into visual objects.

The shape of an object is defined by means of an ? - map. It determines for each pixel whether it belongs to the VO (? - value > 0) or not (? -value = 0). For an opaque object the corresponding ? values are equal to 255, while for a transparent object they range from 1to 254. Most of the reported work on shape coding deals with the efficient coding of binary shapes, with ? -value = 0 being the background and ? - value = 255 being the object.

A.1. Binary Shape Encoding

In the case of binary shape, shape information is divided into 16x16 binary alpha blocks (BABs). These blocks may contain any combination of transparent or opaque shape values. Blocks that are completely transparent or opaque are signaled at the MB level. For blocks that contain both transparent and opaque locations, MPEG-4 defines five additional modes of encoding utilizing a combination of motion compensation and context-based arithmetic encoding (CAE). These modes are signaled using a variable length codeword that is dependent on the coding mode of surrounding MBs, and they are (i) no motion vectors, no shape update; (ii) no motion vectors, shape update (interCAE); (iii) motion vectors, no shape update; (iv) motion vectors, shape update (interCAE); (v) intra-shape (IntraCAE).

INTRA Mode

The first shape coding mode relies completely on an INTRA-VOP CAE. In this approach, the MB is processed in scanline order. A template of 10 pixels is used to define a context for the shape value at the current location. This template is shown in Figure 11(a) and extends two pixels to the left, right and top of the pixel to be encoded. Hence, the context depends on the current MB and previously decoded shape information. Additionally, shape values to the right of the current MB may be unknown. For these values, each undefined pixel is set equal to the value of the closest value within the MB. Having

computed the context for the current pixel, the probability that the location is transparent (or opaque) is determined with a table look-up operation. The table is defined in the MPEG-4 specifiation and contains probabilities for the possible 1024 contexts. Finally, the block is coded using the derived probabilities and an arithmetic code.

INTER Mode

While the INTRA-VOP shape coding method is always available to an encoder, four additional shape coding modes (modes i-iv above) appear in predicted VOPs (e.g., P-, B- and sprite restricted to global motion compensation S(GMC)-VOPs). Here, motion compensation is used to provide an initial estimate of the BAB. To compute this estimate, the encoder signals whether a differential value is included in the bitstream. If included, MVs are recovered by adding the differential to an estimate that is derived from either neighboring BABs or co-located luminance information (selection of the appropriate MV is specified in the standard.) Then, binary shape information from the reference VOP is extracted using pixel accurate motion compensation. These samples can be used for the current BAB. Alternatively, if the encoder signals the presence of an arithmetic code, binary shape information is sent with an INTER-VOP CAE. The INTER-VOP template is shown in Figure 11(b) and contains nine pixel values. Four of these locations correspond to the current BAB, while the other five samples return shape information from the reference VOP. Like the INTRA-VOP case, undefined pixels are set equal to the value of the closest value within the MB, and an arithmetic code is derived using probabilities specified for each of the 512 contexts.

Lossy Encoding

Besides changing the coding modes at the encoder, additional mechanisms are specified for controlling the quality and bit-rate of binary shape information. As a first method, MBs can be encoded at reduced resolution. Accomplished by reducing the resolution by a factor of two or four with a majority operation, the resulting 8x8 or 4x4 BABs are encoded using any of the available compression modes and transmitted to the decoder. After reception, the reduced resolution BAB is decoded and upsampled using an adaptive filter. The filter maps a single shape value to a 2x2 block of shape values. As defined in the MPEG-4 standard, the filter relies on the nine pixels surrounding the low-resolution shape value. When recovering shape data from a 4x4 BAB, the interpolation procedure is applied twice.

Spatial Scalability

Besides reducing the resolution of the BAB, two other options can effect the bit-rate and quality of encoded shape information. First, the efficiency of the CAE depends on the orientation of the shape

information. To increase this efficiency, an encoder may choose to transpose the BAB before encoding. This is signaled to the decoder, which transposes the BAB after applying the appropriate decoding steps. As a second option, spatial scalability is introduced in the second version of MPEG-4 [57]. With this method, both a base layer and enhancement layer are provided to the encoder. The base layer is decoded using previously discussed methods. The enhancement layer refines the shape information of the base layer. This is accomplished by predicting the higher resolution block from either the lower resolution data at the same time instant or higher resolution data in previously enhanced VOPs. If the estimated block deviates from the actual shape data, a CAE can then be used to encode the higher resolution shape.

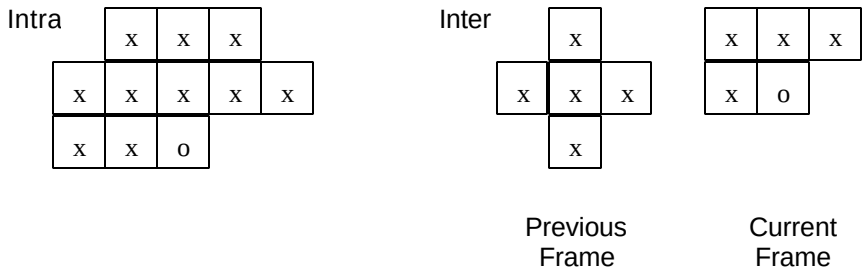

Figure 11: Templates denoting the context for the coded pixel. (a) The INTRA-CAE template. (b) The INTER-CAE template. Alignment is done after motion compensation from the reference VOP.

A.2. Encoding of Grayscale Shape Data

Once the binary shape data has been encoded, grayscale shape data can be sent to the decoder in the form of transparency values. This information consists of four 8x8 blocks that are co-located within each BAB. Encoding the transparency data is almost identical to methods utilized for luminance data. In fact, motion compensation for the transparency information relies on the MVs provided for the luminance channel. The only significant difference between the transparency and luminance data is that overlapped motion compensation is not allowed when sending transparency information.

Two extensions to grayscale coding are included in the second version of MPEG-4 [57]. The first extension is that a bitstream is no longer restricted to one channel of transparency data, and it can now contain up to three channels of grayscale shape data. The encoder signals the presence and content of each auxiliary channel, which can be any combination of transparency, depth, disparity or texture data. Additionally, channels can be denoted as "User Defined" for future extension. The second extension is

the use of the shape-adaptive DCT (SA-DCT). This procedure incorporates the binary shape data into the DCT calculation of the luminance and auxiliary information.

B. Shape Error resilience in MPEG-4

Shape encoding relies on a combination of motion and spatial predictions for efficient representations. Within an error prone environment, this leads to increased sensitivity to channel noise and packet loss. When the error resilience mode is enabled within an MPEG-4 bitstream, modifications in the shape encoder reduce this sensitivity to transmission errors. These changes occur within the computation of the CAE. In non-resilient modes, the context utilized for arithmetic encoding relies on shape data from current and neighboring MBs. To account for potential packet loss, MPEG-4 redefines the context of the CAE when the encoder enables the error resilience mode. This is accomplished by denoting any pixel location that is external to the current video packet as transparent. Applicable to both INTER- and INTRA-CAE modes, this approach limits the propagation of shape errors in a noisy channel.

A second method for protecting shape data within the MPEG-4 framework is data partitioning. This approach separates the MB header, binary shape information and MVs from the texture information. A resynchronization marker (the motion marker) defines the boundary between the two components. The advantages of this approach are twofold. First, an error within the texture data does not effect the decoding of shape. Second, data partitioning facilitates unequal error protection, which can be employed to increase the protection of transmitted motion and shape information from channel errors.

While data partitioning is quite useful for increasing the error resilience of shape coding, it is not applicable to all modes of the shape encoder. Specifically, data partitioning is only defined within the context of binary shape data. When grayscale shape information must be sent, data partitioning cannot be utilized within the MPEG-4 specification. This precludes unequal error protection. Additionally, it inhibits the use of RVLC to protect DCT coefficients within the bitstream, including the texture, transparency or auxiliary channels. In this scenario, errors within a video packet are difficult to localize, and the entire packet must often be discarded.

A final method for error resilience within the MPEG-4 specification is the insertion of a video packet header. This header can appear periodically in the bitstream. When present, it serves as a resynchronization marker and denotes the start of a MB. Additionally, the header contains information for decoding the transmitted MBs, even if previous MBs were lost in the channel. This introduces error resilience for all methods of shape encoding. The video packet header also provides redundant

information from the VOP header. In practice, this allows decoding a VOP even with a corrupt header. Unfortunately, this is only possible when shape information is not present in the bitstream. When shape encoding is employed, the decoder is susceptible to errors within the VOP header, as it contains information about the size and spatial location of the underlying shape. These values can change from VOP to VOP but are not included as redundant information within the video packet header. Thus, errors within a VOP header can result in significant corruption of the decoded frame.

VI. Summary and Concluding Remarks

While Shannon's separation theorem suggests that placing redundancy through channel coding is all that one needs to do, to combat transmission errors, this is not the case under real-time constraints. Here, one has to carefully allocate redundancy between source and channel coding. Furthermore, it is possible for a well-designed decoder to improve the picture quality even upon receiving a damaged bitstream. Finally, there are closed-loop source coding methods that rely on feedback information from the decoder. These methods have been shown to be very effective, but they have a somewhat small range of applications, as feedback channels are unavailable in many scenarios.

In this paper we reviewed encoder ER mechanisms that have already been adopted by recent coding standards, plus additional approaches that we believe are effective and practical. We also discussed decoder-based error concealment techniques and some closed-loop methods. By doing so, we hope to have provided an almost complete overview of the state-of-the-art in low latency, low bitrate video transmission for error prone environments. It is encouraging that the two most recent video coding standards, H.263 and MPEG-4, by including a wide variety of ER tools, can lead to acceptable video quality even in highly error prone environments – something older video coding standards such as H.261 or MPEG1 could not achieve.

This leads us to some speculations about the future of real-time video transmission, video coding, and video coding standards. Over the past 10 years, the achievable bandwidth and QoS over the same type of physical network have been continuously increasing. On the Internet, for example, new access technologies such as ADSL allow for much higher bitrates, and the backbone infrastructure has until now kept pace with those developments. There is no reason why this should change in the future. Higher available bitrates and lower packet loss rates will in turn result in higher picture quality. A similar trend is observable in wireless networks, where technologies that can provide higher bitrates and lower error rates than currently common are driven by one killer application of such systems: Internet access.

Many researchers in the networking community therefore believe that the importance of error resilient source coding will decrease in the future. Simple, transport-based techniques such as FEC or packet duplication might be sufficient to yield quality levels that users can accept at computational demands that they are willing to pay for. A few Internet researchers even suggest that compressed video is entirely obsolete – if the bandwidth is available, why not use it for uncompressed video?

Higher QoS, on the other hand, will inevitably lead to higher user demands of service, which, for video applications, translates to bigger picture sizes and higher reproduced picture quality. For the television industry, HDTV is one of the most important developments since the introduction of color TV. Once the broad public gets used to that resolution and quality, the demand for good quality compressed video will be much higher than it is today. We therefore believe that compressed video in general, and error resilient compressed video in particular will continue to be important research topics. For non-broadcast type of applications through terrestrial or wired channels, we might in the near future see sufficiently short round-trip delay that allow for the use of re-transmission protocols such as TCP. Error resilient coding for such applications may thus become obsolete. However, any form of video broadcast applications such as digital TV and HDTV, as well as any transmission using satellite links where the light speed delay makes re-transmission unfeasible, will continue to rely on error resilient video coding.

Acknowledgement

The authors would like to acknowledge the contribution of J. Brailean of PacketVideo Corp, in the planning phase of this paper.

References

[1] Y. Wang and Q. Zhu, “Error control and concealment for video communication: a review,” *Proceedings of the IEEE*, vol. 86, pp. 974-997. May 1998.

[2] J. D. Villasenor, Y.-Q. Zhang and J. Wen, “Robust video coding algorithms and systems,” *Proceedings of the IEEE*, vol. 87, pp. 1724 -1733. Oct. 1999.

[3] M. R. Banham and J. C. Brailean, ``Video coding standards: error resilience and concealment,'' in *Signal Recovery Techniques for Image and Video Compression and Transmission*, A. K. Katsaggelos and N. P. Galatsanos, editors, ch. 5, pp. 133-174, Kluwer Academic Publishers, 1998

[4] A. K. Katsaggelos, F. Ishtiaq, L. P. Kondi, M.-C. Hong, M. Banham, and J. Brailean, ``Error Resilience and Concealment in Video Coding," *Proc. EUSIPCO-98*, pp. 221-228, Rhodes, Greece, Sept. 8-11, 1998.

[5] M. Budagavi, W. Rabiner Heinzelman, J. Webb, and R. Talluri, “Wireless MPEG-4 video communiction on DSP chips”, *IEEE Signal Processing Magazine*, vol. 17, pp. 36-53, Jan. 2000.

[6] ITU-T Recommendation H.320, "Narrow-band visual telephone systems and terminal equipment", July 1997

[7] ITU-T Recommendation H.221, "Frame structure for a 64 to 1920 kbit/s channel in audiovisual teleservices", July 1997

[8] ISO/IEC IS 13818-1: *Information Technology - Generic Coding of Moving Pictures and Associated Audio Information – Part 1: System*, 1994.

[9] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson: "RTP: A Transport Protocol for Real-Time Applications", RFC1889, January 1996, available from ftp://ftp.isi.edu/in-notes/rfc1889.txt.

[10] ITU-T Recommendation H.323v2, “Packet Based Multimedia Communications Systems,” Mar. 1997.

[11] RFC2543: M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, “SIP”, March 1999, available from ftp://ftp.isi.edu/in-notes/rfc2543.txt.

[12] RFC2327: M. Handley, and V. Jacobson: "SDP: Session Description Protocol", April 1998, available from ftp://ftp.isi.edu/in-notes/rfc2327.txt.

[13] J. C. Bolot, H. Crepin, and A. Vega-Garcia: "Analysis of Audio Packet Loss in the Internet", Proc. Of the 5th International Workshop on Network and Operating System Support for Digital Audio and Video, pp. 163-174, Durham, April 1995.

[14] ITU-T Recommendation H.324, "Terminal for low bit-rate multimedia communication", Feb. 1998.

[15] ITU-T Recommendation H.223, "Multiplexing protocol for low bit rate multimedia communication", Mar. 1996.

[16] Special issue on packet video, *IEEE Trans. on circuit and System for Video Technology*, vol. 3, no. 3, June 1993.

[17] J. Wen and J. Villasenor, "A class of reversible variable length codes for robust iamge and video coding", *Proc. 1997 IEEE International Conf. On Image Proc.*, Santa Barbara, CA Oct. 1997.

[18] J. Wen and J. Villasenor, "Reversible variable length codes for robust image and video transmission". *1997 Asilomar Conference*, Pacific Grove, CA, Nov. 1997.

[19] D. W. Redmill and N. G. Kingsbury, "The EREC: An error resilient technique for coding variable-length blocks of data," *IEEE Trans. Image Proc.*, vol. 5, no. 4, pp. 565-574, Apr. 1996.

[20] P. Haskell and D.Messerschmitt, "Resynchronization of motion compensated video affected by ATM cell loss," in ICASSP-92, San Francisco, CA, USA, Mar. 1992, vol.3, pp. 545-548

[21] N. Naka, S. Adachi, M. Saigusa, and T. Ohya, "Improved error resilience in mobile audio-visua communications," in IEEE International Conference on Universial Personal Communications, Tokyo, Japan, Nov. 1995, vol. 1., pp.702-706

[22] G. Côté, S. Shirani and F. Kossentini: "Optimal mode selection and synchronization for robust video communications over error prone networks" *IEEE Journal on Selected Areas in Communications*, May 1999, Submitted.

[23] S. Wenger, "Video redundancy coding in H.263+," *Proc. AVSPN*, (Aberdeen, UK), Sept. 1997.

[24] S. Wenger, G. Knorr, J. Ott, F. Kossentini: "Error resilience support in H.263+", *IEEE Trans. on circuit and System for Video Technology*, vol. 8, no. 6 pp. 867-877, Nov. 1998.

[25] K. N. Ngan and C. W. Yap, "Combined source-channel video coding," in *Signal Recovery Techniques for Image and Video Compression and Transmission*, A. K. Katsaggelos and N. P. Galatsanos, editors, ch. 9, pp. 269-297, Kluwer Academic Publishers, 1998

[26] L. Kondi, F. Ishtiaq, and A. K. Katsaggelos, "Joint source-channel coding for scalable video," *Proc. 2000 SPIE Conf. On Visual Communications and Image Processing*, San Jose, CA, Jan. 2000.

[27] V. A. Vaishampayan, "Design of multiple description scalar quantizers," *IEEE Trans. Info. Theo.*, vol. 39, pp. 821-834, May 1993.

[28] A. Ingle and V.~A. Vaishampayan, ``DPCM system design for diversity systems with applications to packetized speech," *IEEE Trans. Speech and Audio Processing*, vol. 3, pp.~48-57, Jan. 1995.

[29] Y. Wang, M. Orchard, and A. Reibman, ``Optimal pairwise correlating transforms for multiple description coding", *IEEE Int. Conf. Image Proc. (ICIP98)*, (Chicago, IL), Oct, 1998. Vol. 1, pp. 679-683.

[30] V. K. Goyal, J. Kovacevic, R. Arean and M. Vetterli, ``Multiple description transform coding of images," *IEEE Int. Conf. Image Proc. (ICIP98)*, (Chicago, IL), Oct, 1998. Vol. 1, pp. 674-678.

[31] X. Yang and K. Ramchandran, "Optimal multiple description subband coding," *IEEE Int. Conf. Image Proc. (ICIP98)*, (Chicago, IL), Oct, 1998. Vol. 1, pp. 684-658.

[32] Y. Wang and D. Chung, "Non-hierarchical signal decomposition and maximally smooth reconstruction for wireless video transmission," in Goodman and Raychaudhuri, eds., *Mobile Multimedia Communications*, pp. 285-292. Plenum Press. 1997.

[33] Q. Zhu and Y. Wang, ``Error concealment in visual communications", in A. R. Reibman and M. T. Sun, eds. *Compressed Video over Networks*. Marcel Dekker, Inc. To appear 2000.

[34] A. K. Katsaggelos and N. P. Galatsanos, editors, *Signal Recovery Techniques for Image and Video Compression and Transmission*, Kluwer Academic Publishers, 1998.

[35] S. Aign, ``Error concealment for MPEG-2 video,'' in *Signal Recovery Techniques for Image and Video Compression and Transmission*, A. K. Katsaggelos and N. P. Galatsanos, editors, ch. 8, pp. 235-268, Kluwer Academic Publishers, 1998.

[36] M.C. Hong, L. Kondi, H. Scwab, and A. K. Katsaggelos, "Video Error Concealment Techniques," *Signal Processing: Image Communications*, special issue on *Error Resilient Video*, 14(6-8), pp. 437-492, 1999.

[37] Q.-F. Zhu, Y. Wang, and L. Shaw, "Coding and cell loss recovery for DCT-based packet video," *IEEE Trans. CAS for Video Tech.*, vol. 3, no. 3, pp. 248-258, June 1993.

[38] H. Sun and W. Kwok, "Concealment of damaged block transform coded images using projections onto convex sets," *IEEE Trans. Image Proc.*, vol. 4, no. 4, pp. 470-477, Apr. 1995.

[39] G.-S. Yu, M. M.-K. Liu, and M. W. Marcellin, "POCS-based error concealment for packet video using multiframe overlap information," *IEEE Trans. CAS for Video Tech.*, vol. 8, no. 4, pp. 422-434, Aug. 1998.

[40] H. Sun, K. Challapali, and J. Zdepski, "Error concealment in digital simulcast AD-HDTV decoder," *IEEE Trans. Consumer Electronics*, vol. 38, no. 3, pp. 108-117, Aug. 1992.

[41] W.-M. Lam, A. R. Reibman, and B. Liu, "Recovery of lost or erroneously received motion vectors", *Proc. ICASSP'93*, Minneapolis, Apr. 1993, pp. V417-V420.

[42] A. Narula and J. S. Lim, "Error concealment techniques for an all-digital high-definition television system," *SPIE Visual Commun. And Image Proc.*, Cambridge, MA, 1993, pp. 304-315.

[43] B. Girod and N. Färber: "Feedback-based error control for mobile video transmission", Proc. IEEE, Special Issue on Video for Mobile Multimedia, vol. 87, pp. 1707-1723, Oct. 1999.

[44] W. Wada, "Selective recovery of video packet loss using error concealment," IEEE J. Select. Areas Commun. Vol. 7, pp. 807-814, June 1989.

[45] E. Steinbach, N. Färber, and B. Girod, “Standard compatible extension of H.263 for robust video transmission in mobile environments,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 872-881, Dec. 1997.

[46] Y. Tomita, T. Kimura and T. Ichikawa, “Error resilient modified inter-frame coding system for limited reference picture memories,” *Int. Picture Coding Symp. (PCS),* Berlin, Germany, Sept. 1997, pp.743-748.

[47] ITU-T Recommendation H.263, "Video coding for low bit rate communication", Feb. 1998

[48] S. Wenger, G. Côté, M. Gallant and F. Kossentini: "H.263 Test Model Number 11, Revision 1", Q15-G-16R1, August 1999, available from ftp://standard.pictel.com/video-site

[49] G. Côté, B. Erol, M. Gallant and F. Kossentini: "H.263+: Video coding at low bit rates", *IEEE Trans. on circuit and System for Video Technology*, vol. 8, no. 6 pp 849-866, Nov 1998.

[50] T. Wiegand, N. Färber, and B. Girod, “Error-Resilient Video Transmission Using Long-Term Memory Motion-Compensated Prediction,” to appear in *IEEE Journal on Selected Areas in Communications*, June 2000.

[51] J. C. Brailean and A. K. Katsaggelos, ``Simultaneous Recursive Motion Estimation and Restoration of Noisy and Blurred Image Sequences," *IEEE Trans. Image Processing*, vol. 4, no. 9, pp. 1236-1251, Sept. 1995.

[52] T. Nakai, and Y. Tomita: “Core Experiments on Feedback channel Operation for H.263+” ITU-T SG15 contribution LBC 96-308, November 1996.

[53] J. Wen, J. Villasenor, and H.D. Shin, “Proposal for error resilience in H.26L”, ITU-T documentation Q15-C-36, Eibsee, Germany, 1997.

[54] T. Einarsson, J. Wen, and J. Villasenor, "Proposal for Longer Motion Vectors in H.263+", ITU-T documentation Q15-B-21, Sunriver, USA, 1997.

[55] J. Wen, J. Villasenor and H.D. Shin, “Proposal for single-thread motion vector prediction in H.26L error resilient mode”, ITU-T Documentation Q15-C-35, Eibsee, Germany, 1997.

[56] RFC2429: C. Bormann, L. Cline, G. Deisher, T. Gardos, C. Maciocco, D. Newell, J. Ott, G. Sullivan, S. Wenger, and C. Zhu: "RTP payload format for the 1998 version of ITU-T Rec. H.263 video (H.263+)", RFC2429, May 1999, available from ftp://ftp.isi.edu/in-notes/rfc2429.txt.

[57] "Information technology -- Generic coding of audio-visual objects – Part 2: Visual, Final Proposed Draft Amendment 1," ISO/IEC JTC 1/SC 29/WG 11 N2802 (MPEG-4), July 1999.

[58] J. Wen, J. Villasenor, "Utilizing soft information in decoding of variable length codes", *Proc. 1999 IEEE Data Compression Conference,* Snowbird, UT, Mar. 1999.

[59] M. Bystrom, S. Kaiser, A. Kopansky, "Soft source decoding with applications", Submitted to *IEEE Trans. Circuits and Systems for Video Technology,* Nov. 1999.

[60] A. K. Katsaggelos, L. P. Kondi, F. W. Meier, J. Ostermann and G. M. Schuster, ``MPEG-4 and rate-distortion-based shape-coding techniques," *IEEE Proc.,* special issue *on Multimedia Signal Processing,* vol. 86, no. 6, pp. 1126-1154, June 1998.