Elsevier J. of Visual Communication & Image Representation (Special issue on Emerging H.264/AVC Video Coding Standard), Vol. 17(2), April 2006

Error Resiliency Schemes in H.264/AVC Standard

Sunil Kumar[1*], Liyang Xu[1], Mrinal K. Mandal[2] and Sethuraman Panchanathan[3]

[1]Electrical and Computer Engineering Department, Clarkson University, Potsdam, NY 13699, USA ({skumar, xul}@clarkson.edu)

[2]Electrical and Computer Engineering Department, University of Alberta, Edmonton, Canada T6G 2V4 (mandal@ece.ualberta.ca)

[3]Computer Science and Engineering Department, Arizona State University, Temple, AZ 85287, USA (panch@asu.edu)

* Corresponding author

Abstract

Real-time transmission of video data in network environments, such as wireless and Internet, is a challenging task, as it requires high compression efficiency and network friendly design. H.264/AVC is the newest international video coding standard, jointly developed by groups from ISO/IEC and ITU-T, which aims at achieving improved compression performance and a network-friendly video representation for different types of applications, such as conversational, storage, and streaming. In this paper, we discuss various error resiliency schemes employed by H.264/AVC. The related topics such as non-normative error concealment and network environment are also described. Some experimental results are discussed to show the performance of error resiliency schemes.

Index Terms: H.264/AVC, JVT, MPEG-4 Part 10, video coding standards, video coding, video compression, error resiliency, error-resilient video coding.

I. Introduction

With the success of MPEG-2 in DVD and HDTV applications, the demand for higher coding efficiency in video-based services has grown significantly. In 2001, ITU-T Video Coding Experts Group (VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG) formed the Joint Video Team (JVT) to develop a new video coding standard with better coding efficiency. The committee completed the final draft of ITU H.264 also known as ISO MPEG-4 Part 10 in May 2003. The H.264/AVC (Advanced Video Coding) [1] is thus a new video coding standard of the ITU-T VCEG and ISO/IEC MPEG, which achieves better compression than all other existing video coding standards. The H.264/AVC has been developed based on previous standards such as H.261, MPEG-2 (also known as H.262), H.263 and its enhanced versions H.263+ and H.263++. This new standard is designed to address

applications that are likely to use transmission media such as Cable Modem, xDSL, or UMTS that offer much lower data rates than broadcast channels [2].

Apart from better coding efficiency, the standard has also given strong emphasis to error resiliency and the adaptability to various networks [2]. To give consideration to both coding efficiency and network friendliness, H.264/AVC has adopted a two-layer structure design (Fig. 1): a video coding layer (VCL), which is designed to obtain highly compressed video data, and a network abstraction layer (NAL), which formats the VCL data and adds corresponding header information for adaptation to various transportation protocols or storage media [2]. For stream-based protocols such as H.320, H.324M or MPEG-2, the NAL delivers compressed video data with start codes such that these transport layers and the decoder can robustly and easily identify the structure of bit stream. For packet-based protocols such as RTP/IP and TCP/IP, the NAL delivers the compressed video data in packets without these start codes [3].

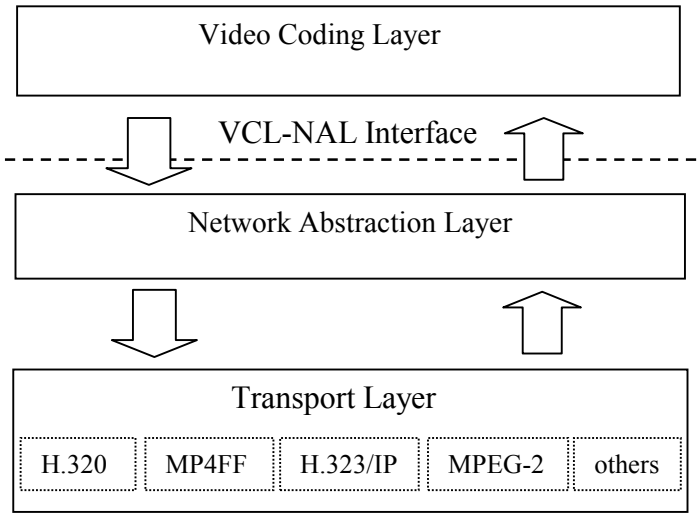


Fig. 1: VCL and NAL layers in H.264/AVC.

A coded video sequence in H.264/AVC consists of a sequence of 'coded pictures'. A coded picture can represent an entire *frame* or a single *field* (for interlaced frame). However, we shall use the term '*frame*' to represent both the entire frame as well as a field, in this paper. The main features of H.264/AVC, which distinguish it from the previous video compression standards, are briefly discussed below [2, 3]:

- Two context adaptive coding schemes, CAVLC (context-adaptive variable-length coding) and CABAC (context-adaptive binary arithmetic coding), improve coding efficiency by adjusting the code tables according to the surrounding information.
- B-frame can be used as reference frame.
- Multiple reference frame motion compensation is allowed, which also improves the prediction accuracy. The restriction that only the immediate previous frame can be used as reference frame is thus removed.

- More motion compensation block sizes and shapes, such as 8x4, 4x8 and 4x4, are supported. The minimum luma motion compensation block size can be as small as 4x4.
- ¼ pixel motion estimation improves prediction accuracy. It has the same prediction accuracy as in MPEG-4 [5], but with lower interpolation complexity.
- Directional spatial prediction is applied in intra-coded macroblocks (MBs) of pictures to reduce the amount of information before their block transform.
- In-loop deblocking filtering removes the blocking artifacts caused by transform and quantization.
- Small block-size transform of 4x4 is used rather than 8x8 used in earlier standards, which results in less ringing artifacts.
- The Discrete Cosine Transform (DCT) is replaced by integer transform that is exact-match inverse transform, thus avoiding drift during inverse transform.
- Parameter sets are used between the encoder and decoder to achieve synchronization in terms of syntax.
- Flexible macroblock ordering (FMO) partitions a frame into different slice groups.
- Data partitioning groups a slice in up to three packets by their importance.
- SP/SI synchronization/switching frames reduce the penalty of switching between ongoing video bitstreams by avoiding transmission of an I-frame.
- Encoder can send redundant representation of some regions of a frame to enhance robustness to data loss.

This paper focuses on the error resiliency schemes adopted in H.264/AVC assuming a typical networking environment. The remaining parts of this paper are organized as follows. General framework of error resilient video coding is discussed in Section II. Section III discusses H.264/AVC error resiliency schemes: semantics and syntax for error detection, data partitioning for unequal error protection (UEP), slice interleaving and flexible macroblock ordering, SP/SI frame for bitstream switching, reference frame selection, parameter sets, error concealment schemes, and intra-block refreshing by R-D control. The salient characteristics, limitations and typical applications of different error resiliency schemes are presented in Section IV. Since video applications depend on networks to transmit the coded video bitstream, the characteristics of common networks, e.g., PSTN, ISDN, Internet and wireless network are briefly discussed as well in Section IV. Some error resilience results are presented in Section V. Finally, the conclusions are given in Section VI.

II. Error Resilient Video Coding

The schematic of a typical video encoder is shown in Fig. 2 [4]. For video coding, a frame is divided into MBs of 16x16 pixels. For each MB, motion estimation finds the best match from the reference frame(s) by minimizing the difference between the current MB and the candidate MBs (from the reference frame). These residual MBs form a residual frame that is essentially the difference between the

current frame and the corresponding motion compensated predicted frame. Simultaneously, motion vectors (MVs) are used to encode the locations of MBs that have been used to each MB in the current frame. The residual frame is then transformed through DCT or integer transform, and quantized. Usually, the quantized coefficients and the MVs are coded by variable length codes (VLCs). The VLCs (e.g., Huffman code, arithmetic code) achieve a higher compression ratio compared to the fixed length codes, and hence the VLC schemes have been extensively used in almost all coding standards to encode various syntax elements. For every coded frame, the encoder transmits the transformed coefficients, motion vectors and some header information essential for decoding. Some frames, known as intra-frame, in the video sequence are coded without using motion estimation/compensation [4].

When the compressed video bitstream is transmitted over a communication channel, it is subjected to channel errors/noise. Generally, forward error correction (FEC) code is used for protecting data against channel errors. The FEC is effective for random errors, but inadequate in the case of long-duration burst errors. Ghandi and Ghanbari [47] have studied the use of hierarchical quadratic amplitude modulation (QAM) for channel modulation. This scheme provides unequal protection to the bits depending on their priority level. The symbols with the same high priority bits are assigned to the same QAM constellation cluster and any two neighboring nodes in a cluster only differ in one bit. This scheme provides a significant PSNR improvement when the channel SNR is lower.

To handle the errors, the following stages are required in an error resilient decoder [4]:

- Error detection and localization
- Resynchronization
- Error concealment

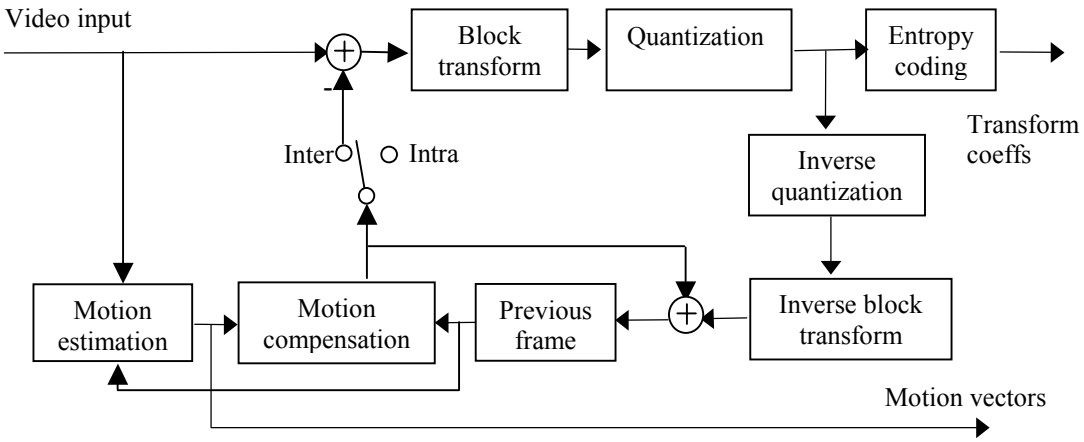


Fig. 2: The block diagram of a typical video encoder [4].

Error detection is done with the help of video syntax and/or semantics. When violation of video semantics/syntax is observed, decoder reports an error, and tries to *resynchronize* at the next start code,

which is typically a long codeword and different from any combination of other possible codes. For some video coding standards (e.g., MPEG-4), which use reversible variable length codes (RVLCs), data is recovered from the corrupted packet by carrying out decoding in backward direction. However, H.264/AVC test model is based on the assumption that the data recovery does not bring a significant advantage to the reconstructed frames [7]. Therefore, the corrupted packets are simply discarded and the lost region of video frame is concealed. The *error concealment* schemes try to minimize the visual artifacts due to errors, and can be grouped into two categories: intra-frame interpolation and inter-frame interpolation. In intra-frame interpolation, the values of missing pixels are estimated from the surrounding pixels of the same frame, without using the temporal information. On the other hand, the inter-frame interpolation is done based on the corresponding region(s) of the reference frame(s). If a motion vector is missing, it can be calculated based on the motion vectors of the surrounding regions. An error concealment scheme suggested for H.264/AVC is discussed in Section III (i).

The choice of error resilient and compression schemes generally requires a tradeoff, and it is difficult to achieve both strong error resiliency and good compression simultaneously. Note that the error resiliency schemes introduce some redundancy in the data. On the other hand, the compression schemes aim to remove various redundancies (e.g., spatial, temporal and statistical) from the data. The spatial, temporal, and statistical redundancies are typically removed by transforms (such as DCT) or predictive techniques (such as DPCM), inter-frame prediction (including motion compensation), and entropy coding, respectively [8]. However, removing the redundant information from the video data makes it more vulnerable to noise. For example when inter-prediction is being employed, a predicted-frame (or P-frame) is dependent on previous reference frame(s). When errors occur in the reference frame, the predicted frame will also degrade due to error propagation. The use of VLCs also makes error propagation more likely in the packet data.

III. Error Resiliency Schemes in H.264/AVC

The error resiliency schemes in H.264/AVC are mainly contained in the VCL. Some of these schemes (such as use of slices, data partitioning, and placement of intra-MBs) have also been used in the previous video coding standards whereas some others (such as parameter sets, redundant slices) are either new or are implemented differently.

a) Semantics, Syntax and Error Detection:

The H.264/AVC video coding standard explicitly defines all the syntax elements, such as motion vectors, block coefficients, picture numbers, and the order they appear in the video bitstream. Syntax actually is the most important tool for ensuring compliance and error detection. Like other video coding standards, H.264/AVC [1] only defines the syntax of the decoder in order to allow flexibility in specific implementations at the encoder. However "it provides no guarantees of end-to-end reproduction quality, as it allows even crude encoding techniques to be considered conforming" [2]. Basically a video bitstream corrupted by error(s) will incur syntax/semantics error(s). Due to the use of VLC, errors often propagate in the bitstream until they are detected. The syntax/semantics errors may include [6]:

1. Illegal value of syntax elements.
2. Illegal sync header.
3. More than 16 coefficients are decoded in a 4x4 block.
4. An incorrect number of stuffing bits are found. This could also occur when extra bits remain after decoding all expected coefficients of the last coded block in a video packet.
5. Some of the coded blocks in a video packet cannot be decoded.

The decoder behavior in the presence of errors has been described in the standard, but the error concealment is not within the scope of H.264/AVC standard. However, there are two recommended non-normative error concealment schemes, which will be discussed later in this section.

b) Data Partitioning:

Since some syntax elements in the bitstream are more important than others, data partitioning enables unequal error protection according to the importance of syntax elements. For example, the header information (e.g., picture size and quantizer) controls the whole slice, a picture or even a whole sequence, whereas loss of transform coefficients only impairs the block to which they belong to or the rest of the slice due to error propagation. Data partitioning of H.264/AVC builds further on H.26L, and allows partitioning of a normal slice in up to three parts (data partition A, B, and C) as discussed below. Each part is then encapsulated into a separate NAL packet [9, 10].

The data partition (DP) *A* contains the header information such as MB types, quantization parameters, and motion vectors, which are more important than the remaining slice data. With the loss of data in DP *A*, data of the other two partitions becomes useless. The DP *B* contains intra coded block patterns (CBPs) and transform coefficients of I-blocks. Because the intra frames and intra-MBs are used as references, the loss of this part will severely impair the recovery of successive frames due to error propagation. The DP *C* contains Inter CBPs and coefficients of P-blocks. Compared to the DPs *A* and *B*, the data contained in DP *C* is less important. However, it is the biggest partition of a coded slice as a large number of frames are coded as p-frames.

Both DPs *B* and *C* depend on DP *A*, but these two are independent of each other in terms of syntax. From this, we can safely draw the conclusion that DP *A* is the most important of the three DPs. Thus in practice, because of the small amount of data in DP *A*, it can be transmitted through out-of-band channel. Compared to MPEG-4 in which all kinds of data are carried by one packet, the DPs in H.264 are more flexible in terms of unequal data protection. To increase the performance further, some researchers have argued for still more flexible data partitioning [11-13] in the H.264 framework. Chen and Ye [11-12] proposed to distinguish low and high frequency transform coefficients and assign low frequency coefficients to DP *A*. Stockhammer [13] proposed to add a copy of header information about intra-MBs to DP *B* so as to make it relatively independent of DP *A*. However, the flexible data partitioning increases the overhead, and it was finally agreed that there was little value in enabling greater flexibility [14].

All partitions should generally be available to start standard-conformant reconstruction at the decoder. However, if the intra or inter partitions are missing due to the loss of DP *B* and/or *C*, the

available header information (e.g., MB types and motion vectors from DP *A*) can still be used to improve the efficiency of error concealment. Table 1 lists the actions at the decoder when some parts of DPs are lost [15]. For example, if DP *C* is lost, the intra information of DP *B* and motion vectors of DP *A* can be used for the concealment. 'Optional' in Table 1 means that the decoder is free to implement the optional concealment mechanisms, whereas the other concealment mechanisms are mandatory for the test model.

Table 1: Recommended actions when partition loss is detected [15].

Available Partition(s)	Concealment Method
A and B	Conceal using MVs from Partition A, and texture from Partition B; intra concealment is optional.
A and C	Conceal using MVs from Partition A and inter info from Partition C; inter texture concealment is optional.
A	Conceal using MVs from Partition A
B and/or C	Drop Partitions B and C. Use motion vectors of the spatially above MB row for each lost MB

c) Slice Structuring: Slice Interleaving/Flexible Macroblock Ordering (FMO):

The H.264 encoder intelligently groups MBs into a slice whose size is less than (or equal to) the size of the *maximum transportation unit* (MTU). Here, the MTU represents the largest size of a packet that can be transported through networks without being split. Prediction beyond the slice boundaries is forbidden to prevent error propagation from intra-frame predictions. The slice structuring strategy thus aims at avoiding error propagation from a corrupted packet to subsequent packets.

The video error concealment schemes perform very well when the lost blocks are arranged in the checker board/scattered blocks fashion or as interleaving of rows [16-18]. This arrangement is helpful in concealing the lost blocks by their surrounding blocks because basically images are smooth at block boundaries [19]. The objective behind the flexible macroblock ordering (FMO) is to scatter possible errors to the whole frame as equally as possible to avoid error accumulation in a limited region. It is well known that as the distance between a corrupted block and the nearest error-free blocks increases, the distortion in recovered blocks grows [20, 21]. Therefore, scattered errors are easily concealed compared to those concentrated in a small region. Experiments reported in [10] showed that, in video conferencing applications with CIF-sized pictures, and at loss rates of up to 10%, the visual impact of the losses can be kept so low that only a trained eye can identify them.

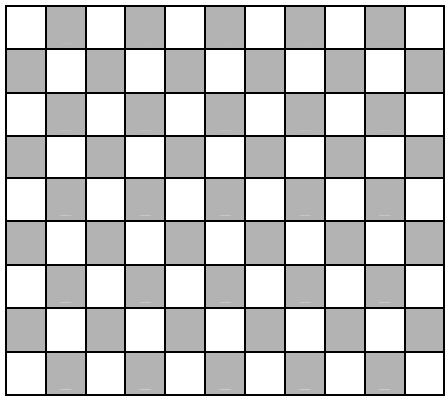


Fig. 3 (a): FMO checker board mode.

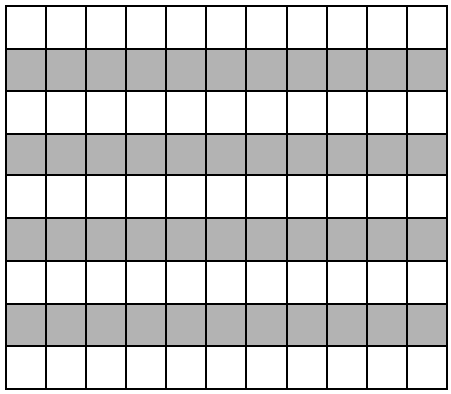


Fig. 3 (b): FMO interleaving mode.

Figs. 3(a) and (b) show the FMO checker board and interleaving modes, respectively, with 2 slices for a frame. The blocks with the same color will be grouped into one slice. When FMO is used, a *macroblock allocation map* (MBA_Map) is added to the parameter set to signal the MB arrangement [22]. The MBA_Map is a data structure that maps the spatial address of an MB to a slice group, in a raster-scan order. In MBA_Map, a number is used for every MB that represents the slice group it belongs to, as shown in Fig. 4 [22].

0	1	0	1	0	1
1	0	1	0	1	0
0	1	0	1	0	1
1	0	1	0	1	0

```
MBA_Map 0,1,0,1,0,1,1,0,1,0,1,0,0,1,0,1,0,1,1,0,1,0,1,0
```

Fig. 4: The MBA_Map of FMO [22].

Note that the MBA_Map supports arbitrary shape such as rectangle, checker board, and interleaving. In slice interleaving, the two neighboring MB rows are separated into different NAL units. It can be implemented with a little change in the MBA_Map. The prediction from left is still available while prediction from upper blocks is forbidden. Normally when a slice is corrupted, decoder can use its upper and lower MB rows corresponding to a different slice to conceal the lost regions. It is better in terms of coding efficiency than checker board mode, but less capable in terms of error recovery. The slice interleaving can be implemented at both VCL and NAL levels [9, 10]. The aforementioned scheme belongs to VCL. At NAL level, encoder can organize slice data by the order of slice interleaving for transmission, but errors may still propagate from an MB row to its neighboring MB row due to prediction.

When the FMO arranges blocks in checker board order, predictions between neighboring blocks are abandoned to cut off error propagation chain, otherwise errors can spread from one slice to another. Unfortunately, overhead bits accompanied with this scheme reduce coding efficiency. The bit rate penalty

in error-free environment highly depends on the picture format, the content, and the quantizer parameter (QP) [23]. The penalty was found to be less than 5% for 6 of the 7 common condition video sequences at a QP of 16, whereas it was about 20% for the worst sequence at a QP of 28 [20].

In order to enhance robustness in the event of loss of slices due to packet drops in network during transmission, H.264/AVC allows the encoder to send *redundant slices* to the decoder. A redundant slice is another representation of one or more MBs in the same bitstream. Note that the decoder needs to be informed when the redundant mode is enabled. A new redundant slice that is predicted from different reference frames is also supported. This increases the possibility that this slice can be reconstructed at the decoder from an error free reference slice [7].

Because intra-frame prediction cannot cross slice boundaries and since the probability of a short packet being corrupted is lower than that for a large packet, the packet loss probability can be reduced if the slices (i.e., transmission packets) are relatively small. Moreover, the error concealment schemes are more effective for short slices. However, the coding efficiency decreases due to reduced intra-frame prediction and the increased NAL overhead [7].

d) SP- /SI- Synchronization/Switching Frame:

Bandwidth scalability in case of pre-encoded sequences can simply be achieved by producing multiple and independent streams of different bandwidth and quality. The server can dynamically switch between the streams to meet variations of the bandwidth available to the client [25]. The SP-/SI mechanism is designed for the purpose of video bitstream switching, but it can also be regarded as an important error resiliency feature in network environments with feedback channel (also known as back channel). Normally, when errors occur in a video bitstream, the synchronization is lost and the recovered frames inevitably degrade until the next I-frame. If the encoder generates an I-frame and transmits it to the decoder for instant error recovery, the generated I-frame will require more bandwidth. However, if the feedback channel is available, decoder can signal the presence of error in the frame(s) to the encoder, and the SP frame can be generated for error recovery.

In the SP-/SI- scheme, multiple video bitstreams of the same source sequence, generated with different parameter sets, are being sent through network. Each decoder uses only one bitstream at a time. When errors occur in its bitstream or when network condition changes, the decoder can switch to another bitstream. Fig. 5 shows a decoder switching from bitstream F_1 to F_2 [24]. The F_1 and F_2 bitstreams are based on different parameter sets, i.e., frames reconstructed from $F_{1,i-1}$ and $F_{2,i-1}$ are different. Here, directly switching from F_1 to F_2 using prediction frame $S_{2,i}$ may incur an unacceptable drift that will propagate to successive frames until an I-frame is encountered. This will cause unpleasant visual artifacts. The prior standards such as MPEG-4 and H.263, forbid switching at the prediction frame. However, H.264/AVC uses SP- frames that enable random switching [24]. In Fig. 5, $S_{2,i}$ is the SP-frame (called primary SP-frame) that is located at the switching point, and $S_{12,i}$ (called secondary SP-frame) is its second representation. $S_{2,i}$ and $S_{12,i}$ have identical reconstructed values of the original frame, but $S_{2,i}$ uses reconstructed frame from $F_{2,i-1}$ as its reference frame whereas $S_{12,i}$ uses reconstructed frame from $F_{1,i-1}$ as

its reference frame. The identical reconstructed values of $S_{2,i}$ and $S_{12,i}$ are primarily obtained by employing appropriate quantization and de-quantization mechanisms at both sides of encoder and decoder, as discussed in [24].

An encoder can also generate a secondary representation known as SI-frame, without using any reference frame. If the bitstream F_1 is corrupted, encoder sends a SI-frame to decoder, which stops the error propagation immediately and switches from bitstream F_1 to F_2. If a SP-frame is used in this scenario, the reference frame of the secondary SP-frame should be correctly received at the decoder.

The SP-/SI- frames are therefore two new picture types adopted in H.264 for regaining synchronization in the video bitstream with only moderate overhead. Besides error recovery, this mechanism can also be used for fast-forward and rewind functions. The SP-/SI-frame mechanism facilitates quick error recovery by enabling switching at random point (i.e., intra as well as predicted frames), and thus increases the reconstructed frame quality. Usually, SP-frames have lower (higher) coding efficiency than P-frames (I-frames). It is important to note however that the SP-frames provide functionalities that are usually achieved only with I-frames [25]. Furthermore, the coding efficiency of SP frames decrease when the feedback delay increases and therefore it may be better to send SI frames when the feedback delay is large.

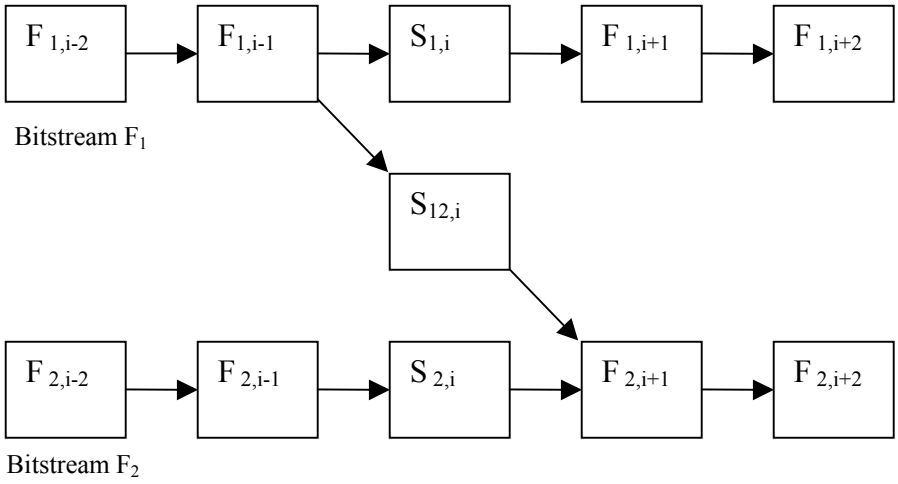


Fig. 5: Switching between two bitstreams using SP-frames [24].

e) Reference Frame Selection:

The reference frame selection mode can be applied both with and without feedback channel. The capacity of reference frame mode is signaled normally by call control protocol, such as H.245 (H.241) in H.323 (H.300). It can be applied to a picture segment instead of the whole picture.

The flexible reference frame selection without feedback channel aims at minimizing the difference between the reference and the predicted regions. On the other hand, the reference frame selection with

feedback channel uses the reference frame based on the feedback acknowledgement as discussed below. There are four possible modes defined in the reference frame mode [26]:

1. NEITHER: no back-channel data is returned from the decoder to the encoder.
2. ACK: the decoder returns only acknowledgment messages.
3. NACK: the decoder returns only non-acknowledgment messages.
4. ACK+NACK: the decoder returns both acknowledgment and non-acknowledgment messages.

In the ACK mode, the encoder uses only an acknowledged segment as a reference for inter-frame encoding. Due to the delay between decoder and encoder, the candidate reference frames are only those frames with ACK, which are several frames before the current frame. Compared to normal scenario, this restriction can impair the accuracy of prediction and thereby introduce extra bits (overhead) that decreased PSNR. In NACK mode, whenever NACK is received for a frame, the new reference frame is selected from amongst the frames coded before the NACK frame. However, if one reference frame is damaged, error propagation and overhead are much higher than the ACK mode because all the frames coded after the reference frame and before the NACK arrives may be affected. The performance of the long-term prediction (used in H.264/AVC standard) may become worse, in this mode.

In the ACK+NACK mode [26], the encoder switches between two kinds of modes according to the upstream messages from the decoder. For instance, the following switching mode can be used.

1. When N NACK messages are continuously transmitted, the mode is switched from NACK to ACK mode.
2. When M ACKs messages are continuously transmitted, the mode is switched from ACK to NACK mode.

Here, the values of N and M are negotiated between the encoder and the decoder. The effect of back-channel and the performance of ACK and NACK modes for H.263 are discussed in [32].

A review of feedback-based error control schemes can be found in [50]. On the other hand, feedback acknowledgement messages should be avoided in applications, such as multicast or broadcast communication. In such cases, 'reference frame selection without back-channel' (e.g., Video Redundancy Coding (VRC)), may be used [27, 28]. VRC divides a bitstream of a sequence into two or more threads. In each thread of P-frames, every frame is coded by using the previous P-frame of the same thread as its reference frame. Any two frames with successive frame numbers are assigned to different threads. All the threads begin and end with a frame called sync-frame. For every specified interval, a sync-frame is inserted into the bitstream to start or end threads. When one frame is corrupted during transmission, the other thread(s) remain intact and at the decoder side the temporal resolution of the reconstructed frames is decreased between two sync-frames. Fig. 6 shows a VRC scheme with 2 threads, with 4 frames per thread [27].

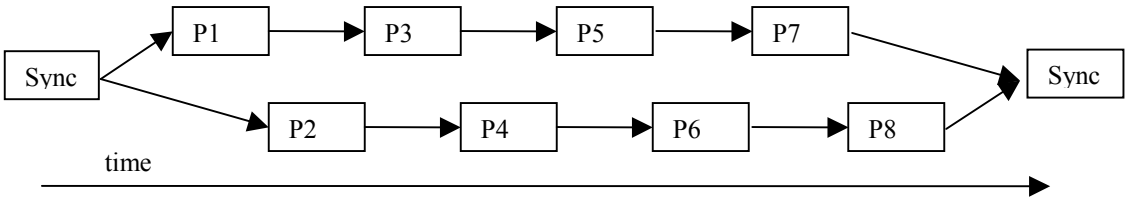


Fig. 6: VRC scheme with 2 threads and 4 frames per thread [27].

f) Parameter Sets:

To correctly decode a video bitstream, the decoder has to synchronize with the encoder, in terms of packets as well as syntax. For example, a decoder needs to know the picture size, entropy coding method, motion vector resolution, and so on. These parameters, which are used for a group of frames or for a series of slices, usually do not change frequently [2]. The H.264 encoder and decoder keep identical parameter tables in their memory that store various possible combinations of these parameters, known as ParameterSets [15]. To signal the parameter configuration to decoder, the index of parameter combination (from tables) is transmitted, instead of the values of parameters themselves. In some cases, the parameter indices can be conveyed out-of-band to give them extra error protection. Since only the indices are transmitted, redundant parameter indices can also be sent with very little overhead for better error resiliency. It also provides more security against intrusion, as the actual parameter values are not transmitted, thus allowing weaker protection of the media stream in a secure system. The structure and transmission of parameter sets on the NAL is shown in Fig. 7 [15].

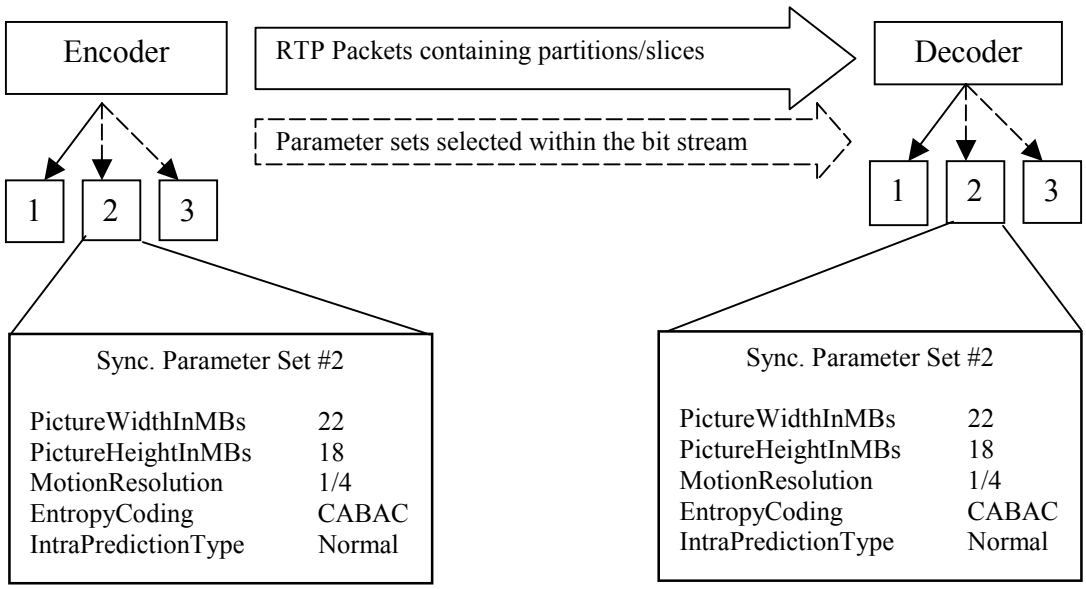


Fig. 7: An example of ParameterSets [15].

g) Intra Block Refreshing by R-D Control:

The error propagation and drift due to predictive coding can be eliminated by periodically inserting intra-coded MBs in the bitstream. This scheme is known as AIR (adaptive intra refresh) in MPEG-4. Unlike periodic or random intra refresh in earlier standards (e.g., MPEG-4, H.263), H.264 uses intelligent intra-block refreshing by R-D (rate-distortion) control, as described below, such that an appropriate block coding option o^* will be selected that minimizes the Lagrangian cost function [7, 10],

$$o^* = \arg \min_{o \in X} (D(o) + \lambda R(o)).$$

Here, o is a block coding mode (inter/intra mode and block size) selected from the set of X coding modes as defined in [32]. The $D(o)$ is the distortion introduced by encoding with mode o, $R(o)$ is the corresponding coding rate, and λ is the Lagrange parameter for appropriate weighting of rate and distortion. $D(o)$ is computed by the SAD (sum of absolute difference) in low complexity mode and by SSD (sum of squared difference) in high complexity mode [32]. For intra-block mode, $R(o)$ represents the coding rate of block coefficients, whereas for inter-block mode it represents the block residual and corresponding motion vector(s). It may be noted that there are more than one motion vectors for each MB except in inter 16X16 mode.

In [10], six codec pairs that use different error resilience features are compared in terms of PSNR values for the packet drop rates from 0% to 20%. The simulation results show that intra block refreshing is an error resiliency scheme as efficient as the other four schemes, i.e., slicing, slice interleaving, data partitioning and FMO. Since the intra-block refreshing scheme is implemented in the encoder, it does not introduce any decoding overhead and can be easily combined with other concealment schemes.

For real-time and conversational video applications, it is not advisable to insert I-frames due to bit-rate constraint and the resulting long delay. Therefore, intra block refreshing is very important to remove artifacts caused by error and inter-prediction drift. However, it is not effective for errors caused by intra-frame prediction.

h) Instantaneous/Gradual Refreshing:

The *instantaneous/gradual refreshing* [29] is typically regarded as a tool to support random access (e.g., fast-forward and rewind), but it is also useful for error recovery when the encoder and decoder lose synchronization due to errors. The MPEG-1 and MPEG-2 standards support random access by using the GOP (Group Of Picture) structure, which starts with an I-frame and contains a number of P- and/or B-frames. A similar structure is also used in H.264 for random access and error recovery. There are two types of strategies for refreshing [29]:

In *instantaneous refreshing*, the frame just before the error-corrupted frames is frozen until the next decodable I-/SI- frame is received. The frozen frame is instantaneously replaced when the refreshment frame arrives. This refreshing procedure also resynchronizes the video bitstream. The *gradual refreshing* can also begin at P-frames. When the reference frame is unavailable, it is set to grey (Y=Cr=Cb=128). The refreshing procedure thus moves on gradually until an I-/SI- frame arrives.

The instantaneous refreshing is a type of assured decoding in which the decoder refreshes a frame only after the indicated conditions are fulfilled [29]. On the other hand, gradual refreshing is a type of best-effort decoding that gives approximate results until the indicated conditions are fulfilled. Note that the support for the refreshing scheme is purely decoder based and does not affect the bitstream. Therefore, it is not a normative part of the H.264/AVC standard.

i) Error Concealment Schemes:

Error concealment is very important for an error resilient decoder. Typically, a decoder utilizes the spatial, spectral and/or temporal redundancies of the received video data to perform error concealment [49]. Most error concealment schemes assume the pixel values to be smooth across the boundary of the lost and retained regions in spatial, spectral and/or temporal domains. To recover lost data with the smoothness assumption, interpolation or optimization based on certain objective functions is often used [16-18, 49, 53-59]. The error-concealment schemes usually reconstruct the lost video data by making use of certain *a priori* knowledge about the video content. Chen and Chen [52] recently proposed an error concealment scheme, based on spatial smoothness, which builds the *a priori* knowledge by modeling the statistics of the video content explicitly, typically in the region of interest (ROI). Context-based models are trained with the correctly received video data and then used to replenish the lost video data. Trained models capture the statistics of the video content and thus reconstruct the lost video data better than reconstruction by heuristics. Zhu *et al*. [53] used second-order derivatives as the smoothness measure, which reduces the blur across the edge while enforcing the smoothness along the edge. Zeng and Liu [54] performed directional interpolation based on the neighbor's geometric structure. Valente *et al.* [55] proposed an improved quadrilinear border interpolation algorithm and a strategy using a spatial activity criterion to efficiently combine several spatial interpolations to avoid the blur on edges. Lam *et al.* [56] proposed a boundary matching algorithm (BMA) to estimate lost motion vectors. Lee *et al.* [57] proposed an affine transform based method to estimate the motion of lost data more accurately. Yan and Ng [58] presented a selective motion vector matching (SMVM) algorithm for MV recovery, which incorporates the status flags of the neighboring pixels of the missing block(s) and constructs new MV sets for block matching. Zheng and Chau [59] used the Lagrange interpolation formula to constitute a polynomial that describes the motion tendency of MVs, which are next to the lost MV.

The specific schemes suggested for the H.264/AVC standard in [30, 31] involve intra and inter picture interpolations. These two concealment schemes can, in fact, be used as benchmarks to evaluate other error concealment schemes. The intra-frame concealment scheme uses interpolation based on weighted average of boundary pixels as shown in Fig. 8. A lost pixel is deduced from boundary pixels of adjacent blocks. If there are at least two error-free blocks available in the spatial neighborhood, only those blocks are used in interpolation. Otherwise the surrounding "concealed" blocks should be used.

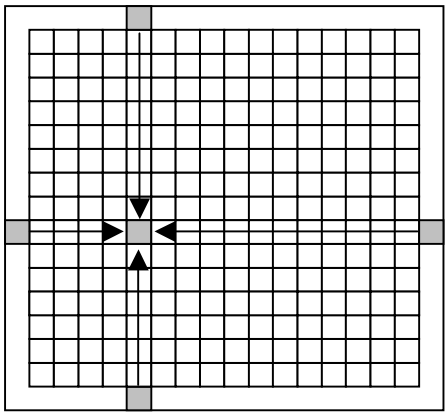


Fig. 8: Spatial concealment based on weighted pixel averaging in a 16X16 block [30].

For inter-frame interpolation based concealment, the recovery of lost MV(s) is critical. Like spatial concealment scheme, the MV interpolation exploits the close correlation between the lost block and its spatial neighbors. Since, the motion of a small area is usually consistent, it is reasonable to predict the MV of a block from MVs of its neighboring blocks. However, the median or averaging over all neighbors' MVs does not necessarily give better results [30]. Therefore, the motion activity of the correctly received slice is first computed. If the average motion is less than a threshold (i.e., ¼ pixel), the lost block will be concealed by directly copying the co-located block from the reference frame; otherwise the MV recovery is done using the procedure shown in Fig. 9 [30]. Note that the selected MV should result in the minimum luminance change across block boundary, when the corresponding block of the previous frame replaces the lost block of the current frame. The decision about selecting an MV, from amongst the MVs of the surrounding blocks, is based on the following equation [30]:

$$\min_{dir \in \{top, bot, left, right\}} \arg \; d_{sm} = \left\{ \sum_{j=1}^{N} | \hat{Y}(\mathbf{mv}^{dir})_j^{IN} - Y_j^{OUT} | / N \right\}$$

where, d_{sm} represents SAD difference between the pixels (of the luminance frame) from the boundaries of lost area and the neighboring boundaries of surrounding blocks [30]. Here $\hat{Y}$ and Y represent the pixel values of the previous and current frame, respectively. The MV^{top1} and MV^{top2} are the MVs of two 8x8 blocks of the upper neighboring MBs. MV^{left}, MV^{right} and MV^{bot} are MVs of the left, right and lower neighboring MBs, in that order. The motion of an 8x8 block is calculated as the average of the motion of the spatially corresponding 4x4 or other shaped (e.g. , 4x8) blocks. The selected MV gives the smoothest boundary transition.

The error concealment schemes introduced above are decoder-based. If a feedback channel is available, a decoder can request the encoder to send out the lost data (or the data in the error propagation region) for error concealment. A few error concealment schemes based on the encoder and decoder interaction have been discussed in [49].

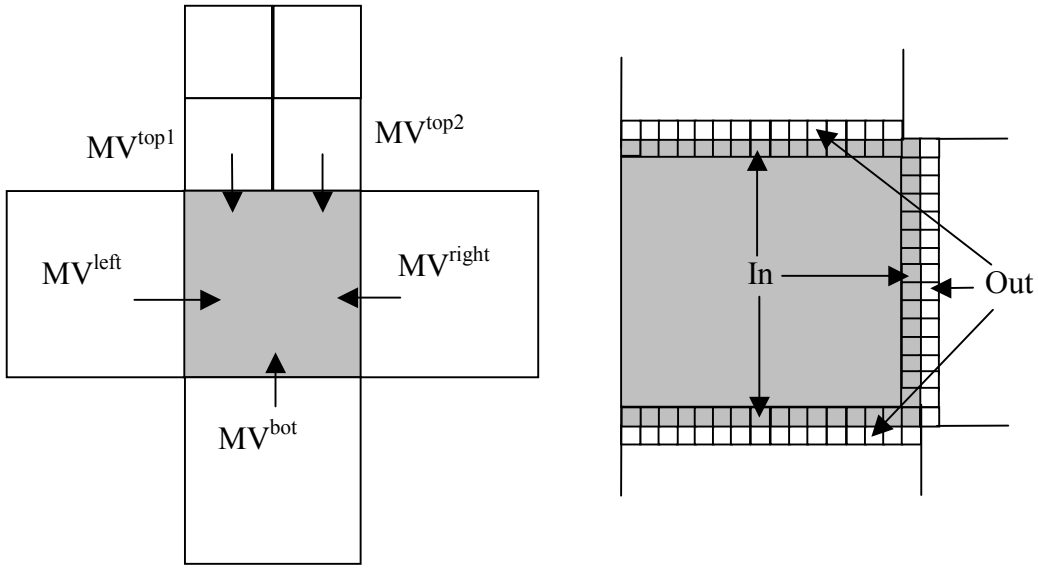


Fig. 9: Estimating the motion vector for prediction [30].

IV. Characteristics, Limitations and Applications of Error Resiliency Schemes

In this section, we first discuss main characteristics and limitations of the H.264/AVC error resiliency schemes. Since the use of these scheme(s) depends on the intended application as well as the network environment, we also provide some discussion about them.

Semantics and syntax are the basic components of H.264/AVC video compression standard. Decoder can detect their violation due to transmission errors and report a decoding error. Normally, the remaining data in the corrupted video packet is discarded. The resynchronization is usually established at the next resynchronization marker. So semantics and syntax can be regarded as inherent error resiliency tools for error detection and resynchronization.

Since syntax elements have different importance, unequal error protection (UEP) for different syntax elements is preferable in an error prone environment. The *data partitioning* (DP) is typically used to enable UEP. The DP can also make decoding more efficient. For example, if an error is detected in DP *A*, the corresponding DP *B* and *C* can simply be discarded. Besides the bit-rate overhead introduced by DP is usually negligible [47].

The use of *I-frame* is undoubtedly the most powerful tool to stop error propagation and recover corrupted pictures. Since the I-frames are independently coded without using temporal prediction, they reset the prediction process. The coding efficiency is therefore lower than the prediction frame. Moreover, transmitting an I-frame requires a large bandwidth (or a long delay on a narrow bandwidth channel) and incurs large bit rate variation, which may not be acceptable in many low-delay applications. On the other

hand, the *intra-block refreshing* provides a reasonable trade-off between the error-resiliency and coding efficiency. It also allows the encoder to maintain a constant bit rate stream, with the help of H.264/AVC integrated rate distortion control mechanism. Additionally, intra-block refreshing is an encoder-based tool, which adds no overhead at the decoder, and can be used with many other error resiliency schemes.

The *reference frame selection* mechanism tries to avoid using a corrupted frame as reference, in order to stop temporal error propagation. It intelligently exploits the feedback acknowledgement information to decide the reference frame. However, the performance of this scheme will degrade for longer network delay, which may reduce coding efficiency (i.e., results in larger residual frame) after motion compensation in ACK mode. Alternatively, it may blindly use a corrupted frame as reference when NACK does not arrive in time. The analysis of the efficiency of feedback-based error resiliency in H.264/AVC can be found in [7].

Slice structuring is a common error resiliency scheme used in many video compression standards. It stops spatial error propagation by limiting the error(s) inside a slice, which forbids any intra-frame prediction across the slice boundary. The use of a smaller slice size will limit error propagation in smaller region, but the coding efficiency will decrease due to higher bit-rate overhead. Slice structuring aids error concealment as well. The use of *FMO*, similar to slice interleaving, aims at spreading the errors due to burst packet losses to the wider region of the frame. This makes the error concealment schemes more effective. But the use of checker board MB arrangement in FMO disallows the intra-frame prediction (to exploit spatial redundancy in neighboring blocks of a frame), which reduces the coding efficiency.

When severe error occurs, decoder may take a long time to achieve synchronization again. In this situation, the use of decoder-based *instantaneous/gradual refreshing* scheme can give acceptable visual effects. If more than one video bitstreams of the same source sequence – generated with different ParameterSets - are being sent through network, a decoder can switch from the corrupted video bitstream to the uncorrupted bitstream using the *SP frame*. A SP-frame is typically more efficient than an I-frame and avoids possible reference drifts. Another type of switching frame, known as *SI-frame*, can also be generated without using any reference frame. It is more robust but less efficient than SP-frame. Note that the SP/SI-frame is a feedback-based scheme that is affected by the longer network delay, that may lower the coding efficiency due to poor reference frame selection.

The use of the *ParameterSets* provides additional error resiliency in H.264/AVC codec. The use of short message (i.e., indices) for codec parameter exchange is good for robustness and information security. Although *Error concealment* is a non-normative error resiliency scheme, it is essential for most error resilient video decoders. Decoder must track the status (corrupted or not, intra or inter MB, and so on) of every MB in a frame. The concealment schemes can be intra-frame and inter-frame. In case of some serious errors (*e.g.* loss of synchronization), error concealment may not be very effective. In such a situation, instantaneous/gradual refreshing can be used to recover the reconstructed frames after the resynchronization has been achieved.

Finally, the choice of one or more of the above-mentioned error resiliency schemes must consider the practical applications as well as network environment. For example, the mobile wireless environment is known to be error-prone due to attenuation, shadowing, fading and multi-user interference. As a result, the channel characteristics vary in time and location. Some aspects about network environment will be discussed later in this section. Here we briefly discuss the role of error resiliency schemes for the three major applications [7].

Circuit-switched and packet-switched *conversational* services (e.g., video-telephony and video-conferencing) require very short end-to-end transmission delay of usually less than 100 ms. Therefore, the encoding, transmission and decoding are performed simultaneously in real-time. As a result, the computational complexity of the codec is very important (especially for hand-held mobile devices) and the error control measures such as interleaving and forward error control (FEC) may not be suitable in these applications. Moreover, the schemes that request the retransmission of corrupted frames cannot be used in these applications due to very low latency tolerance. The error resiliency schemes in this kind of applications have to deal with two problems: minimizing the visual effect of errors within the frame and limiting spatio-temporal error propagation [7]. Intra block refreshing by R-D control, data partitioning, and slice interleaving (FMO, if bandwidth permits) tools may be used with the conversational applications. However, inserting I-frame, reference frame selection based on feedback may not be appropriate.

For Live or pre-encoded packet-switched video *streaming* services (PSS), the bitstream is typically pre-encoded and transmitted on demand through unreliable transmission protocols, making error control (e.g., interleaving and FEC) and error resiliency schemes (as the those used in conversational applications) necessary. Besides, due to the tolerance of relatively long delay compared to conversational applications, feedback-based error resiliency schemes, such as SP-frame and reference frame selection, can be used [10].

In Multimedia messaging services (*MMS*), video data is transmitted using reliable protocols such as *ftp* or *http* that do not require real-time constraints [10]. Therefore, the computational complexity becomes less critical since encoding, transmission, and decoding are totally separate. Error resiliency schemes are not particularly important in this case due to the use of reliable transport protocols.

Different networking environments have different error rate and error resilience properties. The performance of H.264/AVC codec with different error resiliency schemes in the IP and wireless communication environments is discussed in [10] and [7], respectively. Wenger *et al.* [33] have identified three major networking application environments: H.324 for the PSTN, H.320 for the ISDN, and H.323 for the Internet. For each protocol family, the multiplexing and control protocols such as H.223 and H.245, and the corresponding transport layer protocols, are briefly described in this section. We are interested in the error characteristics of data transmission, which actually determines the error resiliency that the video codec should achieve.

H.324 for PSTN: The public switched telephony network (PSTN) is currently the most widely used network that uses point-to-point and circuit-switched communication, with a bandwidth of around 33 Kbps and very small delay [26][35]. The ITU-T H.324 specifies a common method for video, voice and data sharing high speed modem connections [31]. In H.324, H.223 multiplexing protocol mixes various streams of video, audio, data, and the control information into a single bitstream [34]. With the retransmission mechanism that allows retransmission of a lost or corrupted packet with relatively low delay, H.223 provides a practical "error-free" channel within H.324 for video application [27]. 3GPP has adopted H.324 standard [7].

H.320 for ISDN: The video-telephony and videoconferencing systems using the ISDN are the most successful multimedia communication systems in the market today. The ITU-T H.320 uses the ISDN as physical communication platform for video [36, 38]. In H.320, the H.221 defines the transmission frame for audiovisual applications [37]. The H.221 only takes care of the framing structure and does not use error correction because the ISDN works isochronously and the error rate is as low as 10^{-8} [27]. This error rate is too small to justify the use of any error resiliency tools that are inevitably accompanied with some overheads.

H.323 for LANs and Internet: The ITU-T H.323 is for real-time audio, video, and data communications over packet-based networks, such as IP-based Internet and LANs [39]. The H.323 protocol stack uses the Internet Protocol (IP) for inter-network conferencing. In H.323, the H.225 defines the transport/multiplex layer that uses the packet format specified by RTP (Real-time Transport Protocol), and RTCP (Real-time Transport Control Protocol) for logical framing, sequence numbering and error detection. For real-time applications, the RTP based on UDP cooperates with the RTCP to transport multimedia data. For the Internet/LANs, we have to take into account the packet loss rates due to transport error, as discussed in [41].

Wireless Networks for Video Applications: The third generation partnership project (3GPP) uses video codecs to support multimedia applications. Common test conditions of the 3G transmission have been included in H.264/AVC test model [42]. The 3GPP has specified a multimedia telephony service for circuit-switched channels based on ITU-T H.324M. For IP-based packet-switched communication, 3GPP will use the Session Initiation Protocol (SIP) and Session Description Protocol (SDP) for call control and the RTP for media transport. While the H.324 and the RTP/UDP/IP stacks have different roots and switching schemes, there are similar loss and delay effects on the media data in wireless channels [7].

The IEEE 802.11b based wireless LAN is another important network for video applications. The link layer of IEEE 802.11b is not connection-oriented, and therefore a data packet is dropped after a certain number (typically four) of failed attempts. In a single hop communication, the error model is related to random packet loss. However, the random as well as the burst packet loss models should be considered in multihop ad hoc networks. Here, collisions, channel errors and network congestion lead to random packet losses, whereas re-routing delays result in packet loss bursts. Long packet loss burst is a serious problem for video flows [45].

The network/protocol combination characteristics for some networking application environments are summarized in Table 2 [27]. The error resiliency schemes should be chosen depending on the networking application environment. For example, it is not necessary to use strong error resilience for the ISDN, where the bit error rate is less than 10^{-8}. On the contrary, error resiliency schemes such as FMO, RS, and/or reference frame selection may be considered in the Internet-based applications, where typical packet loss rate varies from 5% to 20%.

Table 2: Network/Protocol combination characteristics [27]

Network/Protocol	Bandwidth in Kbps	Bit-error rate	Packet loss rate
PSTN, H.324	<=20 variable, (small packets, e.g. 100 bytes)	low	0% to 5% depending on line quality, QoS and allowed delay
ISDN, H.320	N x 64 fixed, (data transmitted in stream)	$<10^{-8}$	N/A
Internet, H.323 (PSTN modem)	<= 20 variable, (<=1500 bytes per packet)	considers only packet errors	0% ~ 100% (depends on line-quality and backbone-conditions)
Internet, H.323, ISDN dialup or leased line	>= 64 variable, (1500 bytes per packet)	considers only packet errors	0% ~ 100% (depends on backbone-conditions)

V. H.264/AVC Error Resiliency Results

In this section, we discuss the performance of some of the error resiliency tools discussed in Section III at different packet loss rates. Wenger performed experiments to evaluate the performance of a few selected error resiliency schemes in the Internet environment for conversational video applications [10]. The common testing conditions and error patterns follow the description of [43] and [44], respectively. The experiments used the packet loss rates (PLR) of 3%, 5%, 10% and 20%, through an IP/UDP/RTP network with various error resilient modes shown in Table 3. Two test sequences, 'Foreman' and 'Paris', were used whose parameters are listed in Table 4. The other parameters were: a constant QP value, single reference frame, ¼ pel motion resolution, error concealment and no B-slices.

Table 3: Error resilient modes used in the simulation [10]

Experiment	Error resilient mode
Exp 1	One picture, one packet, without using any error resilience tool
Exp 2	One picture, one packet, with intra MB refresh
Exp 3	Slices, typically 2–4 slices per picture were used
Exp 4	Slice interleaving, even and odd numbered MB lines transported in different packets
Exp 5	Data partitioning
Exp 6	Flexible macroblock ordering (FMO)

Table. 4: Sequences used in the simulation [10]

Name	Size	Frame rate	Bit rate	# of frames
Foreman	QCIF (176x144)	7.5 fps	64 Kbps	100
Paris	CIF (352x288)	15 fps	384 Kbps	200

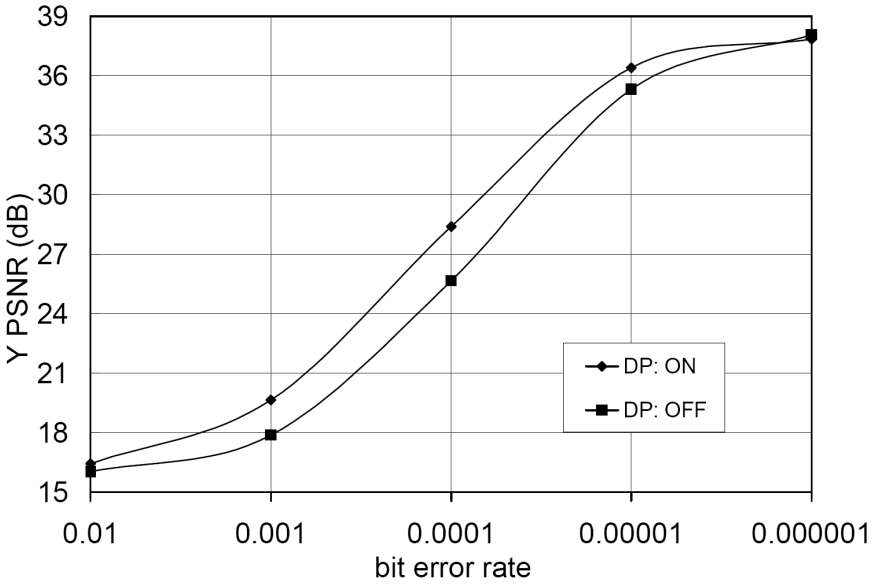
For error free channel, the overheads introduced by the FMO (Exp5) or data partitioning (Exp6) degrade the PSNR of reconstructed sequences by 1 to 2 dB compared to the Exp1 (i.e., the mode without error resiliency scheme). The other three experiments (Exp2-4) have only minor overhead. But the advantage of error resiliency schemes, especially FMO and data partitioning, becomes noticeable as the error rate increases. The quality of video is seriously degraded for packet loss rates above 3% when unprotected H.264/AVC bitstream is used. For 3% PLR, error resiliency tools are not found to be very effective for low bitrate Foreman sequence. However, the PSNR improvement due to these tools for the Paris sequence at bitrate of 384 Kbps is significant. For example, the PSNR improvement in Exp3 is at least 1 dB better than that in Exp1. Similarly, the Exp6 gives the best performance (>9 dB as compared to Exp1) at this error rate. For higher error rates (PLR=5%, 10% or 20%), these error resiliency tools achieve significantly better PSNRs as compared to Exp1 (i.e., without error resiliency tools). For example, there is at least 3.5-6.5 dB difference between Exp1 and others for Foreman sequence. Similarly, the difference ranges from 7-13 dB for Paris sequence. For Paris sequence, different error resiliency tools achieve almost the same performance at 20% PLR. For example, the Exp2-6 provide similar reconstructed PSNR values, which are 13 dB better than that provided by Exp1. For more details about the simulation results, the readers are referred to Figs. 3 and 4 of [10].

Calafate *et al.* tested H.264 error resilience on IEEE 802.11b based multihop ad-hoc networks in [45-46]. The effects of I-frames, random Intra MB refreshing and multiple reference frames were investigated for random packet loss. For burst error evaluation, FMO and random Intra MB refreshing were used. The random error model is very similar to the error model used by S. Wenger in [10], measured by packet loss. For burst error(s), FMOs with 2 or 3 slices per frame are used as error resiliency tools in the experiments. The PSNR and robustness of each frame was tested individually. The experiments showed that FMO is a good option when tuning for error-burst prone network, and intra MB refreshing just 1/3 of the frames is enough to assure a quick convergence.

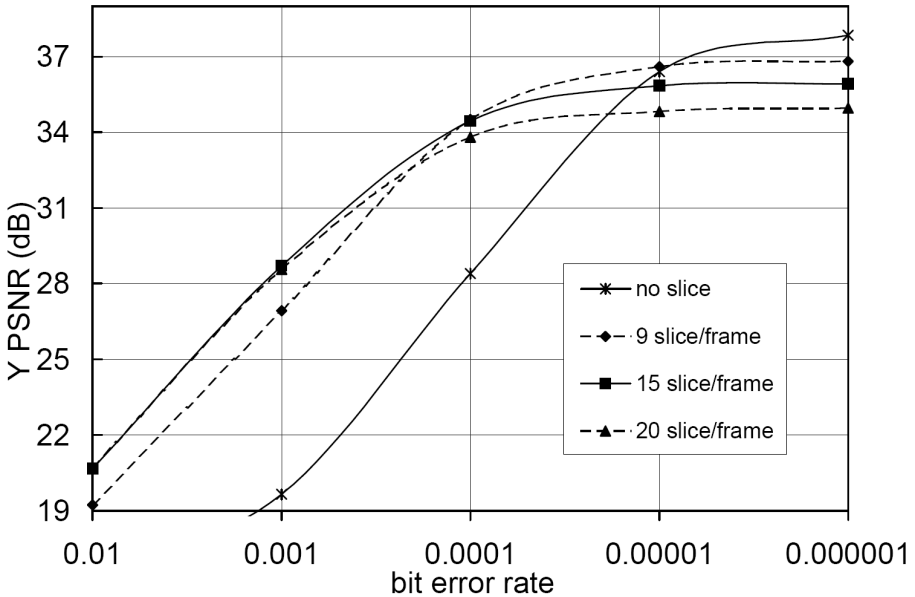
Grangetto *et al.* [48] proposed arithmetic codes (AC) with forbidden symbols as joint coding and error protection tool for robust video transmission. They evaluated their scheme for protecting motion vectors against channel errors in H.264/AVC video packets, and showed that it can provide effective compression and protect error-sensitive motion vector information.

Ghandi and Ghanbari [47] investigated the performance of the DP and slice structuring. The bit-rate overhead introduced by the DP is negligible. Fig. 10 (a) shows that the DP significantly improves the picture quality under moderate BER. But, the benefit of using DP is not distinct when the BER is high (> 10E-2). As shown in Fig. 10 (b), the error resiliency performance of slice structuring is much better

compared to the decoding scheme without slice structuring, even when BER is very high. But the bit-rate overhead introduced by slice structuring significantly degrades the PSNR of reconstructed frames, when BER is very low (e.g., <10E-6). The overhead increases with increase in the number of slices per frame (i.e., smaller slice size).



(a)



(b)

Fig. 10: PSNR vs. bit error rate performance for Foreman QCIF@10Hz, 100Kbps, (a) when DP is enabled and disabled; (b) DP enabled, with and without slices [47].

VI. Conclusions

The H.264/AVC video coding standard aims at achieving improved compression performance and a network-friendly video representation for different types of applications, such as conversational, storage, and streaming. In this paper, we described various error resiliency schemes, including a few non-normative error concealment schemes, employed by H.264/AVC.

The salient characteristics and limitations of these schemes were discussed in detail. The applications and error characteristics in various network environments (e.g., PSTN, ISDN, Internet, wireless) were also briefly discussed. Some experimental results were presented to show the performance of these error resiliency schemes. In practice, the selection of different coding profiles and error resiliency scheme should take into account the intended applications and network characteristics, because they govern the bitrate, error rate and error types.

Acknowledgement: The authors wish to thank Mr. Amit Trivedi, an undergraduate student at Clarkson University, for his help in preparing this paper.

References

1. "Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264/ISO/IEC 14 496-10 AVC," in Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVTG050r1, 2003.
2. T. Wiegand, G. J. Sullivan, G. Bjøntegaard and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Cir. Syst. Video Technol.*, vol. 13, pp. 560–576, July 2003.
3. J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer and T. Wedi, "Video coding with H.264/AVC: tools, performance and complexity," *IEEE Cir. Syst. for Video Technol. Mag.*, Vol. 4(1), pp.7-28, 2004
4. R. Talluri, "Error-resilient video coding in the ISO MPEG-4 standard", *IEEE Comm. Mag.,* Vol 36, pp. 112-119 Jun. 1998.
5. "Information technology -- Coding of audio-visual objects -- Part 2: Visual", ISO/IEC 14496-2, 2000.
6. S. Kumar and L. Xu, "RVLC decoding scheme for improved data recovery in MPEG-4 video coding standard," *Real Time Imaging Journal*, Special issue on Low Bit-Rate Multimedia Communication, Vol. 10, Issue 5, pp. 315-323, Oct 2004.
7. T. Stockhammer, M. M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments," *IEEE Trans. Circuits Syst. Video Technol.*, vol.13, pp. 657–673, July 2003.
8. Cliff Reader, "History of Video Compression (Draft)," in JVT-D068, July 2002.
9. S. Wenger, and T. Stockhammer, "An overview on the H.26L NAL concept," in JVT-B028, Feb 2002.
10. S. Wenger, "H.264/AVC over IP," *IEEE Trans. Cir. Syst. Video Technol.*, vol. 13, pp. 645–656, July 2003.
11. Y. Chen and J. C. Ye, "Flexible data partitioning for improved robustness performance," in JVT-C125, May 2002.
12. J. C. Ye and Y. Chen, "Flexible data partitioning mode for streaming video," in JVT-D136, Jul 2002.
13. T. Stockhammer, "Independent data partitions A and B," in JVT-C132, May 2002.

14. G. Sullivan, “Seven steps toward a more robust codec design,” in JVT-C117, May 2002.
15. S. Wenger and T. Stockhammer, “H.26L over IP and H.324 Framework,” ITU-T SG16 Doc. VCEG-N52, 2001.
16. I. S. Shirani, F. Kossentini and R. Ward “An efficient, similarity-based error concealment method for block-based coded images” *Proc. IEEE Intl. Conf. Image Processing*, Vancouver, Canada, Sep. 2000.
17. P. Salama, N. B. Shroff and E. J. Delp, "Error concealment in encoded video," *IEEE J. Sel. Areas Commun.*, Vol. 18(6), pp. 1129 –114, June 2000.
18. Q.-F. Zhu, Y. Wang and L. Shaw, "Coding and cell loss recovery in DCT based packet video," *IEEE Trans. Cir. Syst. Video Technol.*, Special Issue on Packet Video. Vol. 3(3), pp. 248 - 258, June 1993.
19. W.-J. Chu, J.-J. Leou, “Detection and concealment of transmission errors in H.261 images”, *IEEE Trans. Cir. Syst. Video Technol.*, Vol. 8, pp. 74-84, 1998.
20. S. Wenger, and M. Horowitz, “Scattered slices: A new error resilience tool for H.26L,” in JVT-B027, Feb 2002.
21. S. Valente, C. Dufour, F. Groliere and D. Snook, “An efficient error concealment implementation for MPEG-4 video streams,” *IEEE Trans. Consumer Electronics*, vol. 47, pp. 568-578, Aug. 2001.
22. S. Wenger and M. Horowitz, “FMO: flexible macroblock ordering,” in JVT-C089, May 2002.
23. S. Wenger, “Coding performance when not using in-picture prediction,” in JVT-B024, Feb 2002.
24. M. Karczewicz and R. Kurceren, “The SP- and SI-frames design for H.264/AVC,” *IEEE Trans. Cir. Syst. Video Technol.*, vol. 13, pp. 637–644, July 2003.
25. M. Karczewicz and R. Kurceren, “A proposal for SP-frames,” ITU-T SG16 Doc. VCEG-L27r1, 2001.
26. “Video coding for low bit rate communication annex N,” ITU-T, ITU-T Recommendation H.263 Ver. 1, 1995.
27. S. Wenger, G. D. Knorr, J. Ott and F. Kossentini, “Error resilience support in H.263+,” *IEEE Trans. Cir. Syst. Video Technol.*, Vol. 8, pp. 867 -877, Nov. 1998.
28. S. Wenger, “Video redundancy coding in H.263+”, *Workshop on Audio-Visual Services for Packet Networks (aka Packet Video Workshop)*, 1997.
29. M. M. Hannuksela and S. Wenger, “Random access and time information,” in JVT-B109, Feb 2002.
30. V. Varsa and M. M. Hannuksela, “Non-normative error concealment algorithms,” ITU-T SG16 Doc. VCEG-N62, 2001.
31. “Terminal for low bitrate multimedia communication,” *International Telecommunications Union*, Geneva, Switzerland, ITU-T Draft Recommendation H.324, Dec. 1997.
32. T. Wiegand, N. Färber, K. Stuhlmüller, and B. Girod, “Error-resilient video transmission using long-term memory motion-compensated prediction,” *IEEE J. Select. Areas Commun.*, vol. 18, pp. 1050-– 1050-, Dec. 2000.
33. S. Wenger, T. Stockhammer and M. M. Hannuksela, “Identified H.26L applications,” ITU-T SG16 Doc. VCEG-L34d2, 2001.
34. “Multiplexing protocol for low bitrate multimedia communication,” *International Telecommunications Union*, Geneva, Switzerland, ITU-T Recommendation H.223, 1997.
35. S. Wenger, “Common Conditions for Video Performance Evaluation in H.324/M error-prone systems,” ITU-T SG16 Doc. Q15-I-60, 1999.
36. “Narrow-band visual telephone systems and terminal equipment,” *International Telecommunications Union*, Geneva, Switzerland, ITU-T Recommendation H.320, 1997.

37. "Frame structure for a 64 to 1920 kbit/s channel in audiovisual teleservices," *International Telecommunications Union*, Geneva, Switzerland, ITU-T Recommendation H.221, 1997.
38. S. Wenger, "Recommended simulation conditions for H.26L," ITU-T SG16 Doc. Q15-I-62, 1999.
39. "Telephone systems and equipment for local area networks which provide a non-guaranteed quality of service," *International Telecommunications Union*, Geneva, Switzerland, ITU-T Recommendation H.323, Visual 1996.
40. "Media stream packetization and synchronization on nonguaranteed quality of services LANs," *International Telecommunications Union*, Geneva, Switzerland, ITU-T Recommendation H.225.0, 1996.
41. S. Wenger, "Common conditions for the Internet/H.323 case," ITU-T SG16 Doc. Q15-I-61 1999.
42. G. Roth, R. Sjöberg, G. Liebl, T. Stockhammer, V. Varsa and M. Karczewicz, "Common test conditions for RTP/IP over 3GPP/3GPP2," ITU-T SG16 Doc. VCEG-M77, Austin, TX, 2001.
43. S. Wenger, "Common conditions for wire-line, low delay IP/UDP/RTP packet loss resilient testing," ITU-T SG16 Doc. VCEG-N79r1, 2001.
44. S. Wenger, "Error patterns for Internet experiments," ITU-T SG16 Doc. Q15-I-16r1, 1999.
45. C.M. Calafate and M. P. Malumbres, "Testing the H.264 error resilience on wireless ad-hoc networks," *4th EURASIP Conference focused on Video/Image Processing and Multimedia Communications*, Zagreb, Croatia, 2-4 July 2003.
46. C.M. Calafate, M. P. Malumbres and P. Manzoni, "Performance of H.264 compressed video streams over 802.11b based MANETs," International Workshop on Wireless AdHoc Networking, Tokyo, Japan, March 23-26, 2004.
47. M. M. Ghandi and M. Ghanbari, "Layered H.264 video transmission with hierarchical QAM," *Elsevier J. of Visual Communication and Image Representation* (Special issue of H.264/AVC), to appear in 2005.
48. M. Grangetto, E. Magli and G. Olmo, "Robust video transmission over error-prone channels via error correcting arithmetic codes," *IEEE Commun. Letters*, Vol. 7(12), pp. 596-98, 2003.
49. Y. Wang and Q. -F. Zhu, "Error control and concealment for video communication: a review," *Proc. IEEE*, Vol. 86(5), pp. 974-997, 1998.
50. B. Girod and N. Farber, "Feedback-based error control for mobile video transmission," *Proc. IEEE*, Vol. 87(10), pp. 1707-1723, 1999.
51. Y. Wang, S. Wenger, J. Wen, and A. K. Katsaggelos, "Error resilient video coding techniques," *IEEE Signal Proc. Mag.*, Vol. 17, pp. 61-82, July 2000.
52. T. P. Chen and T. Chen, "Second-generation error concealment for video transport over error prone channels", *Wireless Communications and Mobile Computing* (Special Issue on Multimedia over Mobile IP), Oct. 2002.
53. W. Zhu, Y. Wang and Q.-F Zhu, "Second-order derivative-based smoothness measure for error concealment," *IEEE Trans. Cir. Syst. Video Technol.*, Vol. 8(6), pp. 713 -718, 1998
54. W. Zeng and B. Liu, "Geometric structured based error concealment with novel applications in block-based low-bit-rate coding," *IEEE Trans. Cir. Syst. Video Technol.*, Vol. 9(4), pp. 648 -665, 1999.
55. S. Valente, C. Dufour, F. Groliere, and D. Snook, "An efficient error concealment implementation for MPEG-4 video streams," *IEEE Trans. Consum. Electron.*, Vol. 47(3), pp. 568-578, 2001.

56. W. M. Lam, A. R. Reibman, and B. Liu, “Recovery of lost or erroneously received motion vectors," Proc. ICASSP, Vol. 5, pp. 417-420, Mar. 1993.
57. S. Lee, D. Choi, and C. Hwang "Error concealment using affine transform for H.263 coded video transmissions", *IEEE Electron. Lett.*, Vol. 37(3), pp. 218-220, 2001.
58. B. Yan and K.W. Ng, "A Novel Selective Motion Vector Matching Algorithm for Error Concealment in MPEG-4 Video Transmission over Error-Prone Channels", *IEEE Trans. Consum. Electron.*, Vol. 49(4), pp. 1416-1423, 2003.
59. J. Zheng and L.P. Chau, “A motion vector recovery algorithm of H.264 for digital video using Lagrange interpolation”, *IEEE Trans. Broadcast.*, Vol. 49(4), pp. 383-389, 2003.