Chapter 9

Error Resilience Coding

9.1 Introduction

The coding and transmission of compressed video information over existing and future communications networks with non-guaranteed QoS presents many challenges. Media-based error recovery techniques are necessary for a wide range of applications/environments: interactive video over the internet, personal video communications over wireless networks, and video broadcasting over satellite, to name just a few. Thus, error resilient video coding has recently received a lot of attention from researchers in academia and industry alike.

In a noisy or packet lossy environment, video error resilience techniques are necessary due to the nature of compressed video bit streams. For example, standard-based compressed video bitstreams employ Variable Length Codes (VLCs) as means of entropy coding. A single bit error present in VLC coded video data can lead to a loss of synchronization between the encoder and decoder, resulting in the loss of many video blocks. Multiple bit errors, which are usually due to burst channel errors or to packet loss, may lead to the loss of partial or complete video frames, causing error propagation in the temporal dimension. This propagation is a direct result of motion compensation, which is usually used to reduce video temporal redundancies.

In this Chapter, we will present error resilience coding methods that allow video communication in error prone environments. We will first describe the effects of errors on the compressed video bitstreams. Then, we will discuss data recovery techniques applicable to video communication systems. Finally, we will present error resilience coding techniques to counter the effects of spatial and temporal error propagation.

9.2 Effects of errors and error propagation

Video compression usually employs Variable Length Codes (VLC) in order to achieve high compression gains. VLCs are very sensitive to bit errors, a single

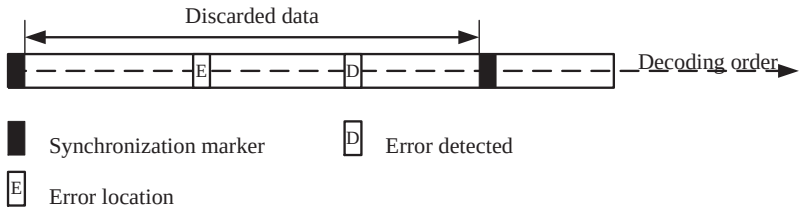

Figure 9.1: Error detection cannot locate the exact location of a bit error when decoding VLCs: data between two synchronization words is usually discarded.

bit error can propagate to many VLCs, and error detection may not be able to locate the exact error location. Moreover, predictive coding, employed in all video coding standards, lend itself to error propagation. In this section, we will discuss the effects of single bit errors and packet losses, and the spatial and temporal error propagation resulting from such errors.

9.2.1 Effects of bit errors

Bit errors occur in bit oriented networks (e. g. wireless networks) and can be detected at the transport level or at the source coding level (see Section 7.2). At the transport level, error detection is normally performed on a block of data. If a video decoder decides to drop a video segment that contains errors, than a bit error is equivalent to a packet loss. If transport level error detection is not available or if the system decides to process known to be incorrect data, bit errors will be present at the video decoder. Single bit errors in the compressed video bitstream will not be localized to a small spatial region, but will lead to the loss of many macroblocks due to the loss of synchronization between the decoder and the bit stream. Since the decoder cannot locate the exact location of the bit errors during VLC decoding, data between two synchronization code words (sync words) is usually dropped, as shown in Figure 9.1. Sync words are uniquely identifiable codewords in the video bitstream that allows the video decoder to regain synchronization. Moreover, decoding of information that could be corrupted by errors lead to annoying artifacts. Rather, error concealment is applied to missing information. We will see later that a limited level of error recovery can be achieved when using different entropy coding methods. When bit errors are dealt with in this fashion, they are equivalent to packet losses, with packet boundaries at sync words locations.

9.2.2 Effects of packet losses

Packet losses occur in packet networks such as ATM or the Internet. Also, bit errors may be seen as packet losses at the video decoder, depending the how such errors are handled, as seen above. Packet sizes, spatio-temporal location of packets within the video sequence and the extent of predictive coding will

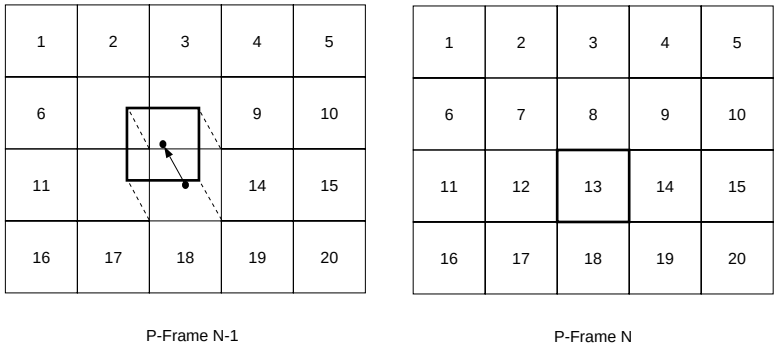

Figure 9.2: Example temporal error propagation due to motion compensation

dictate the error localization of packet losses. A packet loss may affect only a small spatial location of a video frame or can lead to the loss of completed video frame(s), depending on the size of the packet. In some networks, packet losses may occur in a bursty fashion, which could lead to the loss of many video frames. If the packet loss occur in a limited spatial location within a video frame, spatial or spatio-temporal error concealment may be applied to the missing video data. If complete frames are lost, the decoder can choose to freeze the video display with the last correctly received frame, or it can attempt temporal frame interpolation to keep an adequate frame rate. We will discuss later error resilience techniques that improve the localization and minimize the effects of packet losses.

9.2.3 Error propagation in video coding

The nature of predictive video compression systems leads to inevitable spatial and temporal error propagation. Since motion compensation is performed using one or more macroblocks in a previous frame, it can be difficult to assess the error propagation over future frames. The effects of an erroneous macroblock will not only affect the spatially corresponding predicted macroblock but also all macroblocks in which motion compensation is based on this erroneously received macroblock, as shown in Figure 9.2. In this example, macroblock 13 in frame N is predicted by parts of macroblocks 7, 8, 12 and 13 of the preceding N-1 frame. When a feedback channel is available, the encoder can estimate the error propagation based on information about the status of received blocks at the decoder and take action to minimize error propagation due to motion compensation by not using erroneous blocks for prediction. However, when no information is available from the receiver, it is very difficult to assess the error propagation due to errors that occurred in a random and possibly time-varying manner. Until the prediction process based on erroneous data is stopped, the receiver will be unsynchronized with the local decoder in the encoder loop. Prediction errors will occur at the decoder, even if future data is received without

errors, since the prediction frame at the decoder is different than the prediction frame used by the encoder.

9.3 Channel coding techniques for the protection of video bitstreams

This section describes channel coding techniques applied to joint source-channel video coding for networks. More specifically, techniques using Forward Error Correction (FEC) and Automatic Repeat Request (ARQ) in a video source coding algorithm adapted to the network characteristics are discussed.

9.3.1 Forward Error Correction

FEC techniques, widely employed in data communication systems, can effectively be used for single bit error correction of video data. In this open-loop method, the recovery from errors is the responsibility of the decoder. The transmitter adds redundant parity bits which can be used, to an extent, to detect and/or recover lost information. This method also adds a slight amount of delay since the information is grouped into blocks before adding parity information. Recovery from single errors is possible but protection from burst errors or loss of entire packets would require the addition of an excessive amount of parity bits and long blocks of data in order for a typical FEC technique to be effective.

In order to be more effective, FEC should take into account the importance of the different parts of the video bitstreams and unequal error protection should be applied accordingly [1]. The video bitstream is first partitioned into classes of different sensitivities. Important information such as frame size and mode, temporal location of a frame, spatial location indication, coding modes, and quantization parameter have to be protected to ensure a minimum reproduction quality levels. However, errors in different parts of these header bits will lead to significantly different levels of reproduction quality degradation. Some errors will not only degrade local video reproduction quality, but can also cause the decoder to reset due to memory allocation problems. For example, if the encoder changes picture size from QCIF to CIF and this information is not correctly received by the decoder, the decoder may have to reset due to memory management problems. FEC coding can be applied in a cascade manner, where FEC is first applied to all important information, in particular the complete frame and slice headers, and re-applied over information that is increasingly important. More critical information include picture size and coding mode. Since header information is a relatively small part of the bit stream, FEC overhead can be minimal.

9.3.2 Automatic Repeat Request

Automatic Repeat Request (ARQ) techniques are used extensively for channel error recovery in data communications. Several ARQ and Hybrid FEC-ARQ

techniques are employed for robust data communication [2]. ARQ is efficient against burst errors and packet losses but incur additional delays. In this closed-loop error recovery method, a feedback channel from the receiver to the transmitter is maintained. This feedback channel conveys the status of received packets to the transmitter, providing it with the possibility of either re-transmitting the erroneously received packets or containing the effect of their losses. Different ARQ video coding techniques have been proposed in [3, 4].

In these methods, lost or erroneously received video data frames are retransmitted using a feedback channel. Error detection followed by retransmission results in less overhead than FEC alone for the same video reproduction quality [3]. Real-time services require that data-frames are received within a fixed time window, thus ARQ delays should be kept within an acceptable limit. Moreover, source coding algorithms must adapt their rate to the channel conditions to allow retransmission data frames to be sent on a band-limited channel. In a video communication system, an ARQ buffer is present at the transmitting side which contains the data frames which have been transmitted but not yet acknowledge as well as those frames not yet transmitted [3]. The decoder sends acknowledgment (ACK) or negative acknowledgment (NACK) back to the transmitter to inform it of the status of the received data frames. The delay requirements will dictate the data frame loss rate of the ARQ system. It is important to minimize the video frame loss rate, as complete frame losses have a significant impact on video reproduction quality. This can be done with a layered coder, which ensures that the base layer is received through retransmission, and the enhancement layer is only transmitted when the ARQ buffer fullness is under a certain threshold determined by the delay requirements. Moreover, the enhancement layer may never be retransmitted in order to minimize delays, when the network is congested.

Hybrid FEC and ARQ methods are also possible. Two such techniques are type I and type II hybrid ARQ schemes [2]. In type I hybrid ARQ, FEC is applied to all data frames. In type II hybrid ARQ, FEC is only applied when the data frame is retransmitted. Type II hybrid ARQ is obviously more efficient, but also more complex.

ARQ methods may not be possible in applications such as video-on-demand and broadcasting, as a feedback channel is not always available. Thus ARQ is has a limited set of applications.

9.4 Containing the effects of errors in compressed video: example methods

As mentioned earlier, localizing the exact location of bit errors in a compressed video bitstream is not possible, and large amount of possibly correct information must be discarded. In this Section, we describe techniques to contain the effects of errors to small spatial regions without an excessive overhead rate increase.

9.4.1 Synchronization markers

Sync markers are codewords that are uniquely identifiable in the bit stream. Therefore, a VLC, a combination of VLCs, or any other codeword combination in the bitstream cannot emulate a sync word. Moreover, the sync word must also be detectable in the presence of bit errors. These restrictions puts a limit on the minimum length of the sync word. In H.263, for example, the sync word is 17 bit long and equal to 00000000000000001. In order to localize the errors to a small spatial region, sync words may be inserted at various locations, either at a regular spatial interval in the coded frame, or at a regular bit interval in the bitstream. For example, sync words could be inserted at the beginning of every row of macroblocks (regular spatial interval) or once every 256 bits in the coded bitstream (regular bit interval). When sync words are inserted at a regular bit interval, they will be closer (spatially) in high activity regions, since more bits are necessary to represent these macroblocks. When the decoder detects an error in the bitstream, it discards bits until the next sync word is detected. With sync words at a regular bit interval, the error can be localized to a few macroblocks in the high activity areas. We call the video data between two sync words a *slice*.

In order to contain the errors within a slice, data dependencies across slice boundaries within the video frame should be removed. These include motion vector prediction and predictive quantizer update. In order to remove these dependencies, the encoder modifies the coding strategy at the beginning of a new slice. The motion vector of the first macroblock of a new slice is not predicted. Moreover, to ensure spatial synchronization at the decoder, additional fields are inserted in the bitstream, after the sync word:

1. The Macroblock Address of the first macroblock of the new slice, which allows the decoder to spatially re-synchronize by providing the spatial location of this macroblock.

2. The Quantization Parameter (QP), which represents the quantization step size of the current macroblock and re-synchronizes the prediction of QP.

It is also possible to modify the sync word to provide additional error detection capabilities, instead of relying on the detection of invalid VLCs [5]. This is done by replacing some of the sync word bits by parity bits of blocks of data following the sync word. When searching for synchronization at the decoder, the parity bits for consecutive blocks of data are calculated and compared to the parity bits in the sync word. If an error is detected, the current parity bit along with the remaining bits of the sync marker are compared to the corresponding known sync word bits. If no errors occurred, the remaining parity bits should match the sync word bit and synchronization is obtained. When the receiver is in synchronization with the bitstream, the parity bits are used to verify the integrity of the data. If an error is detected, the decoder conceals the associated macroblocks and looks for the next sync word.

9.4.2 Data partitioning

Data partitioning can be implemented into two different ways. It can organize the coded data into segments with different error sensitivity into higher and lower priority data. The higher priority data is necessary to decode the lower priority data. The video segments can be transmitted on separate channels with different error characteristics or guaranteed QoS. Scalable video coding is an example of this type of data partitioning and was discussed in Chapter 8.

Data partitioning can also be used to separate the different type of data within a video bitstream. Usually, motion vectors and DCT coefficients information are coded together for every macroblock. When using this type of data partitioning, motion vector and DCT coefficient information within a packet are respectively grouped together and separated by a boundary marker. The boundary marker is a uniquely decodable codeword that signals the end of the motion vector information and the beginning of the DCT coefficient information. Errors can then be localized to a certain type of data, and the unaffected information can be employed in the decoder. For example, when an error occur in the DCT data but the motion vector data is not affected, this information can still be used to perform motion compensation. It also allows better error detection capabilities. If undetected errors occur in the video packet, the received data can be considered invalid if the boundary marker is not detected. However, if the boundary marker is detected but the macroblock address is incorrect after detecting the next sync word at the start of the next slice, the decoder can assume that the motion vector information was correct and discard the DCT coefficients information.

9.5 Spatial error resilience coding techniques

Once bit errors have been contained to a spatial location within a video frame, it is possible to recover additional information within this corrupted video segment using different methods, as described in this section. The most popular methods are robust entropy coding techniques that provide better synchronization and error localization properties than VLCs. Reversible Variable Length Code (RVLC), the Error Resilient Entropy Code (EREC) and Fixed Length Code (FLC) are example methods discussed here. Secondly, multiple description coding techniques that allows recovery of data from partially received video segments are discussed.

9.5.1 Error resilient entropy coding

Entropy coding is a lossless process that maps source information into a sequence of symbols from a given alphabet. The average length of the output sequence is the measure of the compression efficiency of the coding scheme. The smaller the output length, the better the compression. Optimal performance in a noiseless environment can be achieved by a complete Huffman tree. However, in a noisy environment, such VLCs are very sensitive to bit errors. If an error ocurr while

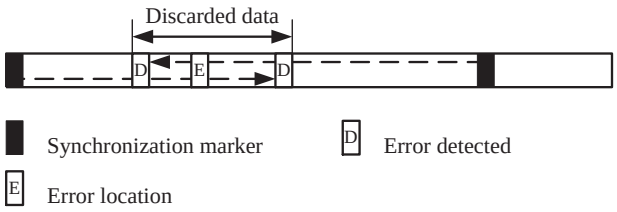

Figure 9.3: Reversible Variable Length Codes enable the recovery of data by allowing decoding in the forward and reverse directions.

decoding VLC data, the decoder loses synchronization with the bit stream, search for the next sync marker, and conceals missing macroblocks between the receive sync words. Therefore, many possibly correct macroblocks will be discarded. In this Section, we present alternative entropy coding methods that are more resilient to bit errors. These methods improve error localization and/or decoder resynchronization.

Reversible Variable Length Codes (RVLC)

RVLCs [13] allow the decoder to recover data by enabling decoding in the forward and reverse directions. RVLCs are VLCs that have the prefix property in the forward and reverse directions, so they can be uniquely decoded in both directions. When the decoder detects an error while decoding in the forward direction, it can look for the next sync marker and start decoding in the reverse direction until an error occurs. Based on the position of the detected errors in the forward and reverse directions, the decoder is able to locate the error within a smaller region in the bitstream as shown in Figure 9.3.

RVLCs can be built in different ways. One method is to take a constant Hamming weight VLC code set and add a fixed-length prefix and suffix to it. The decoder is able to decoder in the forward and reverse directions by counting the number of 1s in the code to detect the end of the code. More complete RVLCs can be generated by increasing the length of the prefix. RVLCs can also be designed by modifying Golomb-Rice, providing good compression efficiency [14].

In general, the advantage of two-way decoding of RVLCs reduces compression efficiency. By proper training on video sequences, the RVLCs are designed to match the probability distribution of the compressed video data, being motion vector information or run-length coded DCT coefficients, such that their compression efficiency loss is minimal.

Also, to fully take advantage of the RVLC's properties, codewords representing the same type of data should be grouped together. Thus RVLCs are usually employed with data partitioning, where instead of successively encoding the motion vectors and DCT coefficient information for each macroblock, motion vectors and DCT information of all the blocks within a slice are respectively

grouped together.

Fixed length codes

Fixed-Length-Codes (FLC), also known as block codes, are less sensitive to error propagation than VLCs [15]. FLC outperforms VLC with synchronization markers in a noisy environment. As opposed to VLC where the input codeword is of fixed length and the output symbols are of variable length, FLC is performed by mapping variable length input codewords to fixed length block codes. An algorithm to construct optimal FLCs, in terms of minimum output bitrate, was proposed by Tunstall [16]. The lower bound on the average length of FLC is the same as VLC, which is the entropy of the source U, $H(U)$, for a discrete memoryless binary source. The upper bound is, however, different. In VLC, the upper bound on the average codeword length is $H(U) + 1$. With Tunstall codes, the output bitrate decreases as the length of the input codewords, N, increases. An important issue with FLC is that the lower bound is limited by the probability of the less likely symbol, p_{min}. As the alphabet size increases, the coding efficiency will decrease. Modified Tunstall codes were first presented in [17], and later in [15] to address this shortcoming.

As for RVLCs, data partitioning should be used with FLCs to improve their coding efficiency. Even though fixed length code allows codeword synchronization, they do not guarantee spatial synchronization and sync codewords are still necessary. For example, if a fixed length code word representing DCT information is corrupted by errors and the resulting codeword alienate a codeword which represents a different spatial location in the video frame, i.e. different DCT block, the following data will be decoded at the wrong location and large areas of the frame will be displaced until synchronization is re-gained by detecting a sync codeword. The FLC codeword assignment can be designed to minimize this type of errors, as proposed in [18].

The EREC system

The EREC system is an error resilient entropy coding technique for coding of variable length blocks of data [19]. The Error Resilient Entropy Code (EREC) is used to convert VLCs to fixed length blocks of data, which are more resilient to bit errors. This allows the decoder to be synchronized with the bitstream at the start of each blocks with minimal additional redundancy.

The EREC frame structure is composed of N slots of length s_i bits for a total length of $T = \sum_{i=1}^{N} s_i$ bits per EREC frame. It is assumed that the values s_i, T, and N are known by the decoder. The slot lengths s_i can be predefined as a function of T. The number of EREC slots N can usually be fixed in advance, and, thus, only T needs to be transmitted to the decoder on highly protected channels. The EREC places the VLC blocks of data of length b_i into the EREC frame structure using a bit reorganization structure that relies on the ability to determine the end of each variable-length block. Figure 9.4 shows an example of the reorganization algorithm. The first stage of the algorithm is to allocate

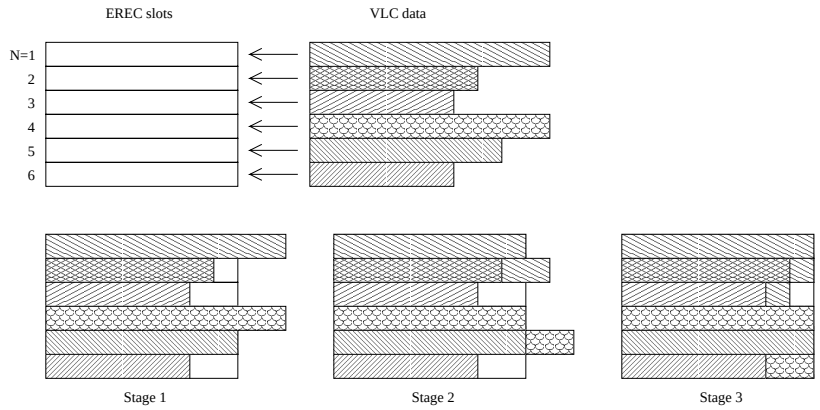

Figure 9.4: Example of the EREC bit reorganizing procedure

each block of variable length data to a corresponding EREC slot. Blocks with an equal number of bits to those available in the slot are coded completely, leaving the slot full. Blocks with $b_i < s_i$ are coded completely, leaving the slot with $s_i - b_i$ bits unused. Blocks with $b_i > s_i$ have s_i bits coded to fill the slot and leave $b_i - s_i$ bits to be allocated. In the subsequent stages, each block of data still to be coded are allocated to the slots with remaining spaces. At stage n, the VLC code at slot i searches slot $i + \phi_n mod N$, where ϕ_n is a predefine offset sequence. Provided that enough bits are available in the set of N slots, all the bits are placed within N stages of the algorithm. In the example in Figure 9.4, the offset ϕ_n is constant and equal to 1. The remaining spaces in the EREC slots are filled with redundant bits. In the absence of channel errors, the decoder follows this algorithm by decoding the variable-length block data until it detects the end of each VLC block, at which point the remaining contents of the slot corresponds to data from parts of other VLC blocks placed later in the algorithm. A pseudo-random offset sequence ϕ_n was shown to provide better error resilience properties [19], because of its uncorrelated nature, i. e. $\phi_{n+k} - \phi_n$ is independent of n. Thus two VLC blocks k slots apart will not search in the same order.

It was shown in [19] that the EREC system provides a relatively large improvement in reproduction quality when applied to DCT-based still image and video codecs compared to VLC with synchronization words. In [20], a joint source-channel coder using the EREC system and H.263 video coding was proposed, with a significant improvement over VLC coding. It is also important to note that the EREC system requires that some information be transmitted on highly protected channels and that, as for fixed-length coding, EREC does not guarantee image/video frame spatial synchronization. Therefore synchronization markers should still be employed.

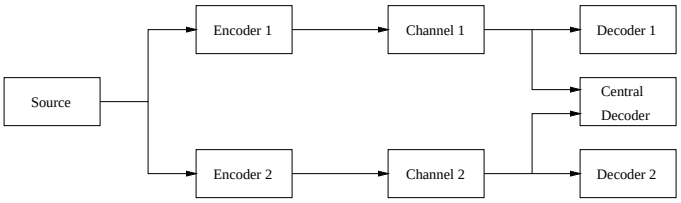

Figure 9.5: Multiple Description Problem

9.5.2 Multiple Description Coding

Multiple Description (MD) coding allow a video decoder to extract meaningful information from a subset of the bitstream. An example would be SNR scalability, which allows meaningful information to be retrieved when only the base layer is received. In order to make use of the enhancement layers, the base layer must be correctly received, which is assured by priority classes that must be supported in the network. On the other hand, MD coders have the ability to produce descriptions that can be equally important (balanced descriptions) and that are also mutually refinable, i.e. receiving two descriptions together and combining them is better than receiving the descriptions separately. Figure 9.5 shows a block diagram of the MD setup. An encoder produces two descriptions that are transmitted over two channels. The encoder has no knowledge on the status of the channels. Usually, the two channel model is employed to represent two different network packets. At the receiving end, we have three decoders, channel 1 decoder, channel 2 decoder, and combined channel 1 and 2 decoder. If information from channel 1 or channel 2 only is received, the decoder is able to reconstruct the video with distortion D_1 or D_2, respectively. If information from both channels is received correctly, then the decoder is able to reconstruct the video with distortion D_0. Achievable rates for given distortions D_0, D_1, D_2 were presented for Gaussian sources in [6]. Two descriptions can be twice as good as one when the individual descriptions need not to be very good. However, when the constraint on individual distortions are sufficiently severe, two descriptions are no better than one, as they become the same description. Moreover, the rate overhead incurred by the multiple descriptions will increase if the individual descriptions need to be good.

Multiple Description Scalar Quantizers

Many different methods have been proposed to achieve multiple descriptions. In [7], multiple descriptions are achieved by the design of multiple description scalar quantizers. The input signal x is quantized to yield an integer index $l = q(x)$, where $q(\cdot)$ represents the quantizer mapping of a uniform threshold quantizer with central bin index l. Information about l is mapped to a pair of indexes $(i, j) = a(l)$ and the index i is transmitted on channel 1 while the index j is transmitted on channel 2. If only channel 1 or 2 works, distortions

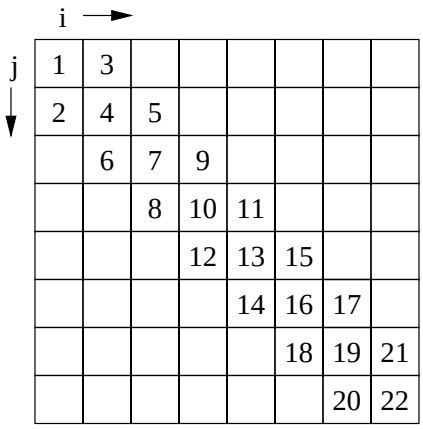

Figure 9.6: Example of multiple description quantizer assignment for a 22 level quantizer.

D_1 and D_2 incur, respectively. If both channels work, the index pair (i, j) is mapped back to the central index l to reconstruct x with distortion D_0. Figure 9.6 presents an example of MD quantizer mapping. If only one index is received, it is possible to estimate the central index l by choosing the central index in the row/column of the received index. Details of index assignments can be found in [7]. This technique as been used for image coding and intra block coding in a video framework [8]. Also, it has been extended to a differential pulse code modulation (DPCM) system, which was applied to a motion compensated video codec in [].

Multiple Description Transforms

Another proposed technique to obtain multiple descriptions is to use transforms that introduce a controlled level of correlation between the two descriptions [9]. The MD objective is realized by forcing dependencies between the two transmitted bit streams, allowing either bitstream to be estimated from the other when one is lost. The transformed based MD approach codes a pair of input variables A and B by forming and coding a pair of transformed variables C and D:

$$\begin{bmatrix} C \\ D \end{bmatrix} = \mathbf{T} \begin{bmatrix} A \\ B \end{bmatrix}. \tag{9.1}$$

Redundancy is added by controlling the correlation in the pair (C, D) of transformed coefficients. The one-channel distortion is reduced because the correlation between C and D allows the information from the lost channel to be estimated from information on the received channel.

For example, transform MD can be applied to video frames as follows. Uncorrelated quantized DCT coefficients of spatially separated blocks are paired

as input variables (A, B). Correlating transforms are applied to each pairs, then entropy coding is performed on the correlated variables. Frequencies are grouped together and a correlating transform T is designed for each groups. As they are the most important, DC coefficient pairs would be coded with high redundancy (correlation).

Generalized Multiple Description Coding

The MD techniques described above are specific to the two channel multiple description problem. When applied to packet network communications with possibly many packets per image, using more than two descriptions would increase performance. This problem of generalized multiple description coding (GMDC) was addressed in [10], and later applied to images using correlating transforms in [11, 12].

9.6 Temporal error resilience coding techniques

This section describes methods to minimize the temporal propagation of errors. As seen above, error propagation due to motion compensation can be difficult to evaluate and such errors can propagate over a large spatial location and over many frames. In this section, we present methods to minimize errors caused by prediction and describe techniques that minimize temporal error propagation.

9.6.1 Intra frame refresh

A simple technique to avoid error propagation in the temporal direction is to increase the frequency of intra frames, which resets the prediction process. Intra frames have no temporal dependencies with other frames while inter frames are predicted, achieving much better compression. While low bit rates are achieved by eliminating spatial and temporal redundancies, error resilience techniques must also exploit both spatial and temporal redundancies. Therefore, the use of intra frames as means of error recovery should be minimized. Moreover, the substantially larger intra frames will add undesirable high latency in a low bit rate environment.

9.6.2 Random intra macroblock refresh

Instead of coding a complete video frame in the intra mode, it is preferable to intra code only some macroblocks within a frame. Using advanced error concealment techniques, errors in many macroblocks can efficiently be concealed. Therefore, randomly intra coding all macroblocks in a frame would not be efficient. One simple technique is to choose an intra coding pattern where only macroblocks that contain texture information (i. e. excluding macroblocks that are skipped or only motion compensated) are intra coded. Therefore, most of the active regions and/or regions of interest will then be intra updated.

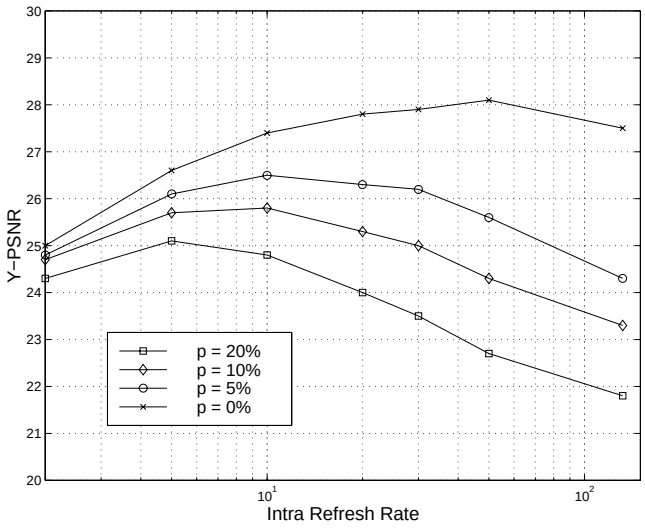

Figure 9.7: Performance of intra macroblock refresh with picture loss rate for PARIS.

Randomly updating macroblocks using the intra coding mode has been suggested in [23, 24]. In [23], Haskell proposes that the portion of macroblocks to be intra updated in a coded frame is chosen based on the life expectancy of the errors. He states that this life expectancy depends on the intra macroblock refresh rate but is fairly independent of the probability p that a macroblock is in error. Also, a method is proposed to only update blocks with high activity (variance).

A relationship between the probability that a macroblock is corrupted by errors, p, and the intra updating frequency, I_{freq}, can be obtained based on experimental results. To obtain this relationship, different video sequences are encoded with a fixed bit rate and the intra refresh rate is varied over four different macroblock loss rates $p = 0\%$, 5%, 10% and 20%. Decoder Y-PSNR results are presented for the sequence PARIS encoded at 64 kbps in Figure 9.7. It can be observed that the frequency of the intra updating can be approximated by

$$I_{freq} = \frac{1}{p}. \tag{9.2}$$

For example, for a probability of macroblock loss p of 20%, each candidate coded macroblock should be randomly updated once in every 5 coded frames.

9.6.3 Rate-distortion optimized macroblock refresh

RD optimized video coding provides an efficient means for coding mode selection. A summary of research work on RD optimized mode selection for an error

free environment can be found in [25]. Here, three coding modes are considered: *skip*, *inter*, and *intra*. The skip mode is a special case of inter mode where no information is transmitted, and the block is simply repeated from the spatially corresponding block in the previous frame. Independently for every block, we choose the mode that minimizes the Lagrangian given by

$$J = D + \lambda R, \tag{9.3}$$

that is, the coding mode that yields the best RD tradeoffs for the block. Using

$$\lambda = 0.85 \times \left(\frac{Q}{2}\right)^2 \tag{9.4}$$

has been shown to provide good RD tradeoffs [25], where Q is the quantization step size of the block.

Using the above method, the coding mode selection is only optimal if the video bitstream is received without errors at the decoder. When errors are present, temporal prediction will allow errors to propagate if the inter mode is chosen. Using the intra coding mode will stop error propagation, but at a higher coding rate cost.

If we know the error concealment method employed by the decoder and error rates of the network, we can achieve better tradeoffs between compression efficiency and error resilience. First, we can attribute the distortion to two sources: distortion D_q caused by the quantization errors, and distortion D_c caused by errors due to prediction from a concealed block. Assuming a block error probability of p, we minimize the Lagrangian

$$J = (1-p)D_q + pD_c + \lambda R. \tag{9.5}$$

Here, the rate R is the rate at which the coded sequence is transmitted, and is the same as in Equation (9.3).

When minimizing the above Lagrangian, we assume that the block considered will be received by the decoder. For this block, two distortions are computed for all three coding modes considered: the coding distortion D_q and the concealment distortion D_c. Then D_q is weighted by the probability $(1-p)$ that this block is predicted from a correct block, and D_c is weighted by the probability p that the same block is predicted from a concealed block. Using this above minimization, good RD tradeoffs can be achieved subject to the probability of error rate and concealment constraints. The error concealment method will directly affect the mode decision. A better error concealment method than the one employed here will yield better RD performance given the same probability of error rate.

Note that, by minimizing Equation (9.5), blocks that are compensated from well-concealed ones in the previous frame will most probably not be coded in the intra mode. If a corresponding block has been perfectly concealed, then $D_c = D_q$ and Equations (9.3) and (9.5) are therefore equivalent

9.6.4 Selective intra refresh based on feedback information

When a feedback channel is available, the status of received macroblocks can be signaled to the encoder, and the encoding mode decision can be adapted accordingly [26]. In this method, macroblocks are coded in intra mode only if they are lost and considered to deteriorate video quality given the motion compensation error propagation and the error concealment method employed.

The receiver operates in the following manner:

1. Lost macroblocks are detected by the network or by the video decoder.
2. Damaged block(s) is (are) specified by calculating the receiver data structure.
3. Notice of lost block(s) with their address(es) is sent back to the transmitter.
4. Error concealment is carried out over the damaged blocks.
5. Decoding resumes.

The received data structure is calculated based on the packetization structure. Usually, all blocks between two sync words will be considered lost if any part of the bitstream between those sync words is corrupted by errors.

After receiving the feedback information, which includes notice of packet loss with the address(es) of damaged block(s), the encoder responds by adapting its encoding strategy in one of the two following manner: Method A:

1. The affected picture area in the local-decoded frames is calculated from the point of the damaged block(s) up to the currently encoded frame by considering error propagation due to motion compensation.
2. Encoding is continued without using the affected picture area (coding the affected blocks in intra mode for example).

Method B:

1. The same error concealment that is performed in the decoder is carried out in the local-decoded frame.
2. Local decoding is re-executed from the point of the concealed block(s) up to the current block using the concealed block(s), so that the encoder decoding loop is the same as the receiver's.
3. Encoding is continued normally throughout the above sequence.

Method B is efficient for short round trip delay (delay between sending the video data at the encoder and receiving the receiver's notice of lost blocks). However, complexity increases as the round trip delay increases. Method A is

less complex, as only the propagation tree needs to be stored at the encoder. Coding efficiency deteriorates by a small amount with method A.

The error propagation tree at the encoder is constructed as follows. As encoding proceeds, a table entry is kept for each block, which includes address(es) of block(s) for which pixel(s) were used for its prediction (due to motion compensation). More sophisticated error tracking methods at the encoder are possible [22]. In this method, the affected blocks at the encoder are weighted by their portion that is used for motion compensation of the lost block. Tradeoffs between coding efficiency and error resilience can be achieved by adapting the number of affected blocks to be updated based on their effect on motion compensated error propagation.

9.6.5 Use of multiple prediction options

Temporal error resilience can also be achieved by coding in *inter* mode, but using a predictor that was not affected by errors. In this mode of operation, a video encoder can use a different picture than the previously received one for prediction and signal to the decoder which picture was used for prediction. This can be done two different ways. In the first method, the decoder can signal to the encoder the status of a received video segment and the encoder can choose not to use this segment for future prediction. We call this mode of operation Reference Picture Selection. Note that this mode requires the use of a feedback channel to signal the status of the received video segments. In the second method, the encoder may add additional picture segments that are predicted from different reference frames, and interleave the prediction process between frames, containing the temporal errors to a small interval. We call this method Video Redundancy Coding. We discuss these methods next.

Reference picture selection

Reference Picture Selection (RPS) dynamically replaces reference pictures in the encoder in response to an acknowledgment signal from the decoder [27]. This can be done with a positive acknowledgment (ACK) or a negative acknowledgment (NACK). In the first case, the decoder sends a ACK message to the encoder every time a picture segment is received correctly. The encoder replaces the reference picture according to the returned ACK. If an error occurs, no ACK is sent back and the reference picture remains the same (the reference frame is the last known to be correct picture). This will lead to a coding efficiency loss, since the temporal distance between the current picture segment and the reference frame will increase. However, the coding efficiency is still superior to that of intra coding. Moreover, the reference picture that is not available at the decoder is not used for prediction, so no picture degradation occur. The coding efficiency will also decline as the round trip delay between sending the video data and receiving the ACK message increases. If the round trip delay is longer than the video frame interval, using the ACK mechanism will lead to a

loss of coding efficiency even if no error occurs, since the reference picture will be further apart than necessary from the current picture segment.

In the NACK mode of operation, the decoder send a NACK message to the encoder when a picture segment has not been received correctly and the encoder adjusts the reference picture accordingly. Once a NACK message is sent by the decoder, the decoding process for this picture segment is stopped until it is received correctly. This will lead to a temporary freezing of the picture, which do not occur in the ACK mode of operation. However, if no error occur, coding efficiency is not affected.

Both these methods are sensitive to errors in the feedback channel. In the ACK case, if an error in the feedback channel occurs, the reference picture is not replaced, leading to a decrease in coding efficiency, but no serious picture quality degradation at the decoder. In the NACK case, if an error occur in the feedback channel, the encoder will not use the correct reference picture, until a NACK is correctly received, leading to a significant picture degradation at the decoder.

It has been found that the NACK method is effective for low error rates while the ACK method is effective for high error rates. When errors in the forward and back channel are correlated, the ACK method outperforms the NACK method [27].

Video Redundancy Coding

Video Redundancy Coding (VRC) improves temporal error resilience using multiple prediction options without a feedback channel [28]. Redundant information is added by inserting additional coded frames predicted from different reference frames and by employing different prediction threads. At least two prediction threads are employed, where temporal predictive coding within a thread is independent of the other threads, as shown in Figure 9.8, where n threads and m frames per thread are used. Coded frames within a thread are predicted from previous frames in the same thread, where the temporal reference of the previous frames are interleaved between threads. All threads start from a common *sync* frame, which can be an intra frame of a predicted frame employing the same reference frame in all threads. If one thread is damaged (due to a packet loss for example), prediction in other threads is not affected, and resynchronization of the damaged thread will occur at the next sync-frame. The decoding process of the damaged thread is usually stopped, in order to avoid annoying temporal error propagation. If a complete video frame is considered as one packet and $n = 2$, then the loss of a video packet will result in a reduction in frame rate by half, until the next sync-frame. The choice of the number of threads and the number of frames per thread depends on the expected network characteristics. Many shorter threads (with fewer frames per threads) is preferable for high packet loss rates, at a lower coding efficiency when errors do not occur, since the distance between the coded frame and it's reference frame will increase as the number of threads increases. VRC with $n = 3, m = 3$ provides good video quality for a picture loss rate of 20% [28].

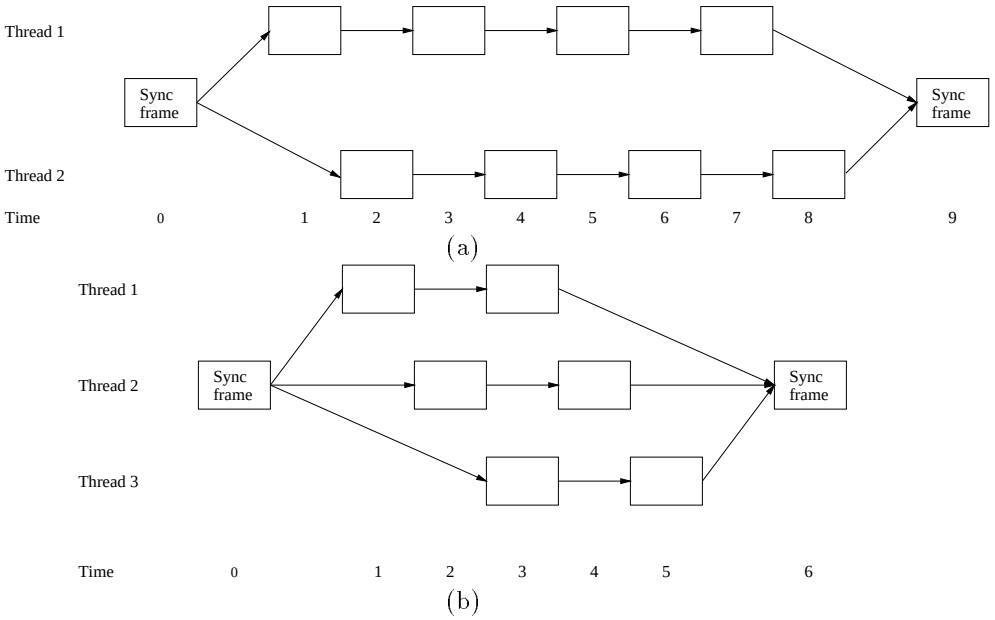

Figure 9.8: Example of thread setup for Video Redundancy Coding: (a) $n = 2, m = 5$, (b) $n = 3, m = 3$.

9.6.6 Error resilience tools in H.263 Version 2

9.6.7 Error resilience tools in MPEG-4

MPEG-4 employs RLVCs for the coding of DCT coefficients. RVLCs are used along with data partitioning, which allows for the grouping of the DCT information, thus taking advantage of the properties of RVLCs. The standard does not specify how RVLC should be decoded, but guidelines are provided as to how data is is recovered using RVLCs [29].

9.7 Further Reading

Bibliography

[1] A. Garzelli A. Andreadis, G. Benelli and S. Susini, "FEC coding for H.263 compatible video transmission," in *International Conference on Image Processing*, Santa Barbara, CA, Oct. 1997.

[2] D. Costello S. Lin and M. Miller, "Automatic-repeat-request error-control schemes," *IEEE Communications Magazine*, vol. 22, no. 12, pp. 5–17, Dec. 1984.

[3] E. Dubois M. Khansari, A. Jalali and P. Mermelstein, "Low bit-rate video transmission over fading channels for wireless microcellular systems," *IEEE Trans. on Circuits and Syst. Video Technol.*, vol. 6, pp. 1–11, Feb. 1996.

[4] P. Cherriman and L. Hanzo, "Programable H.263-based wireless video transceivers for interference-limited environments," *IEEE Trans. on Circuits and Syst. Video Technol.*, vol. 8, no. 3, pp. 275–286, June 98.

[5] W.S. Lee, M.R. Pickering, M.R. Frater and J.F Arnold, "Error resilience in video and multiplexing layers for very low bit-rate video coding systems," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 9, Dec. 1997.

[6] A. El Gamal and T. Cover, "Achievable rates for multiple descriptions," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 6, pp. 851–857, Nov. 1982.

[7] V. Vaishampayan, "Design of multiple description scalar quantizers," *IEEE Trans. on Information Theory*, vol. 39, no. 3, pp. 821–834, May 1993.

[8] M.R. Pickering W.S. Lee, M.R. Frater and J.F. Arnold, "A diversity-based scheme for reducing error propagation in video," in *International Conference on Image Processing*, Santa Barbara, CA, Oct. 1997, vol. 3, pp. 582–585.

[9] V. Vaishampayan M. Orchard, Y. Wang and A. Reibman, "Redundancy rate distortion analysis of multiple description image coding using pairwise correlating transforms," in *International Conference on Image Processing*, Santa Barbara, CA, USA, Oct. 1997.

[10] V. Goyal and J. Kovacević, "Optimal multiple description transform coding of Gaussian vectors," in *IEEE Data Compression Conference*, Snowbird, UT, USA, Mar. 1998, pp. 388–397.

[11] M. Orchard Y. Wang and A. Reibman, "Optimal pairwise correlating transforms for multiple description coding," in *International Conference on Image Processing*, Chicago, Illinois, USA, Oct. 1998.

[12] R. Arean V. Goyal, J. Kovacević and M. Vetterli, "Multiple description transform cosing of images," in *International Conference on Image Processing*, Chicago, Illinois, USA, Oct. 1998.

[13] Y. Takishima, M. Wada, and H. Murakami, "Reversible variable lenght codes," *IEEE Trans. on Communications*, vol. 43, no. 2, pp. 158–162, Feb. 1995.

[14] J. Wen and J.D. Villasenor, "A class of reversible variable length codes for robust image and video coding," in *International Conference on Image Processing*, Santa Barbara, CA, Oct. 1997.

[15] R. Llados-Bernaus and R. Stevenson, "Fixed length entropy coding for robust video compression," in *International Conference on Image Processing*, Santa Barbara, CA, USA, Oct. 1997.

[16] B. P. Tunstall, *Synthesis of noiseless compression codes*, Ph. D. Thesis, Georgia Institute of Technology, 1968.

[17] T. Algra, "Fast and efficient variable-to-fixed-length coding algorithm," *Electronics Letters*, vol. 28, no. 15, pp. 1399–1401, July 1992.

[18] R. Llados-Bernaus and R. Stevenson, "Codeword asigment for fixed-length entropy coded video streams," in *IEEE Data Compression Conference*, Snowbird, UT, USA, Mar. 1998.

[19] D.W. Redmill and N.G. Kingsbury, "The EREC: An error resilient technique for coding variable-length blocks of data," *IEEE Trans. on Image Processing*, vol. 5, pp. 565–574, Apr. 1996.

[20] K.N. Ngan and C.W. Yap, "Combined source-channel video coding," in *Signal Recovery Techniques for Image and Video Compression and Transmission*. Kluwer Academic Publishers, Boston, 1998.

[21] F. Eryurtlu A.H. Sadka and A.M. Kondoz, "Improved performance H.263 under erroneous transmission conditions," *IEE Electronics Letters*, , no. 19970086, Oct. 1996.

[22] N. Faerber E. Steinbach and B. Girod, "Standard compatible extension of H.263 for robust video transmission in mobile environment," *IEEE Trans. on Circuits and Syst. Video Technol.*, Feb. 1998.

[23] 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.

[24] N. Naka, S. Adachi, M. Saigusa, and T. Ohya, "Improved error resilience in mobile audio-visual communications," in *IEEE International Conference on Universal Personal Communications*, Tokyo, JAPAN, Nov. 1995, vol. 1, pp. 702–706.

[25] G.J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Proc. Magazine*, Nov. 1998.

[26] M. Wada, "Selective recovery of video packet loss using error concealment," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 5, pp. 807–814, June 1989.

[27] S. Fukunaga, T. Nakai and H. Inoue, "Error resilient video coding by dynamic replacing of reference pictures," in *IEEE Global Telecommunications Conference*, New York, NY, USA, Nov. 1996, vol. 3, pp. 1503–1508.

[28] S. Wenger, "Video redundancy coding in H.263+," in *Audio-Visual Services over Packet Networks*, Scotland, UK, Sept. 1997.

[29] R. Talluri, "Error resilient video coding in the MPEG-4 standard," *IEEE Comm. Magazine*, vol. 26, no. 6, pp. 112–119, June 1998.