Chapter 2

Background

In this chapter, we'll first point out the applications and transmission characteristics for wireless video applications. Since H.264/AVC video coding standard [1] [2] and the Robust Fine Granularity Scalability (RFGS) [8] are adopted as our platform, a brief introduction of these two techniques is provided. Finally, we give an overview of tools of error resilience and error concealment in current research.

2.1 Video in Wireless Networks

There are three major service categories in wireless networks [3]:

(1) Packet-switched conversational services (PCS) for video telephony and conferencing;

(2) Live or pre-recorded video packet-switched streaming services (PSS); and

(3) Video in multimedia messaging services (MMS).

The transmission requirements for these three different types of applications can be differentiated with respect to the requested data rate, the maximum allowed end-to-end delay, and the maximum delay jitter. This fact results in different system architectures for different services as shown in Fig. 2.1. For example, MMS does not include any real-time constraints, so the processing of encoding, transporting, and decoding can be separated. In PSS applications, we may separate encoding and transmission, but the decoding process is usually started during transmission in order to minimize the initial delay and memory usage in receivers. As to conversational applications, real-time service must be ensured. In these applications, encoding, transmission and decoding are performed simultaneously.

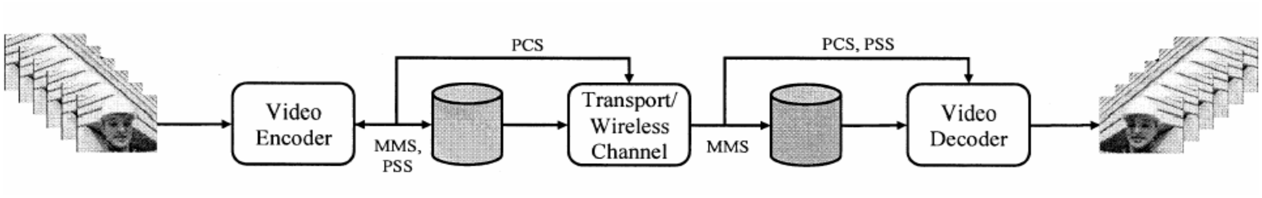


Fig. 2.1 Wireless video application MMS, PSS, and PCS: differentiation by real-time or offline processing for encoding, transmission, and decoding [3]

Although these three service categories require different strategies in encoding, transport, decoding, and the underlying network architecture, video streaming in wireless networks still possess some common characteristics.

First, videos are mostly transmitted in packet-based wireless networks. Fig. 2.2 shows an example of H.264/AVC standard in transport environment. Here, different transport protocols can be used for different purposes. For example, when transmitting video data as a file (as MMS), TCP/IP is suitable. For conversational applications, RTP/UDP/IP would be a good choice. Fig. 2.3 shows a typical packetization of an H.264/AVC NAL unit encapsulated in RTP/UDP/IP.

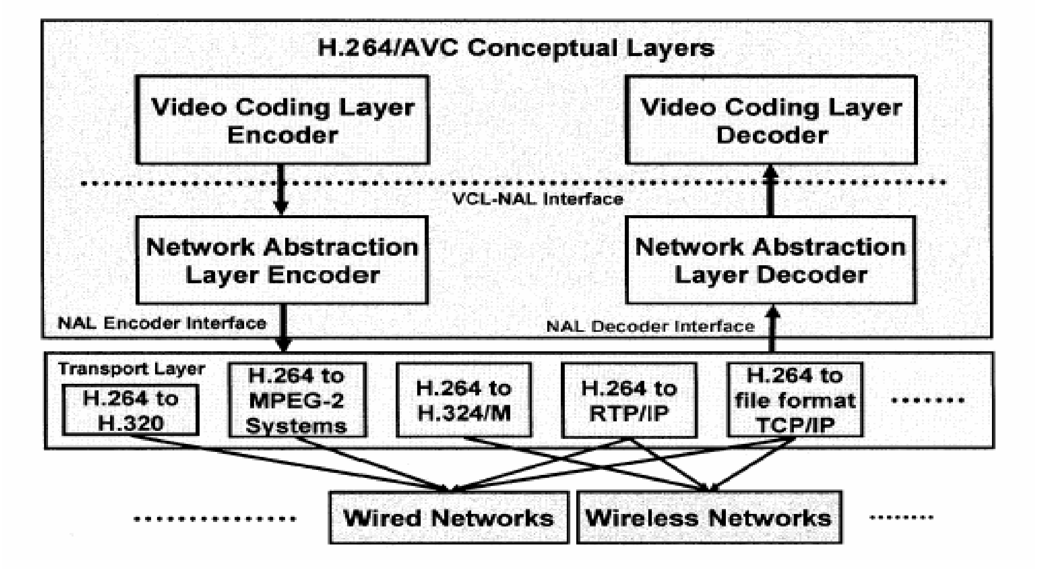


Fig. 2.2 H.264/AVC standard in transport environment [3]

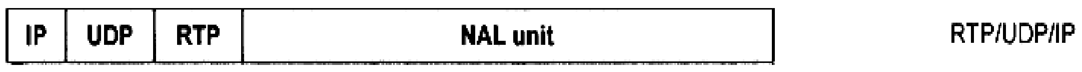


Fig. 2.3 Packetization of an H.264/AVC NAL unit encapsulated in RTP/UDP/IP

Next, the available bandwidth is limited and the transmission cost is expected to be proportional to the bit rate over the communication link. Thus, compression efficiency would be a main requirement if a video coding standard is to be practicable in a wireless environment. This makes H.264/AVC a prime candidate for wireless visual communication because of its superior compression efficiency.

In addition, the wireless environment is characterized by its harsh transmission conditions, like attenuation, shadowing, fading, and multi-user interference. These factors make the wireless environment an error-prone environment. Hence, in the literature, error resilience and error concealment techniques have been widely used and continuously discussed.

As mentioned above, the main characteristics of a wireless network include low bit rates, high bandwidth variations and error prone environment. With its high compression efficiency, H.264/AVC is thought to be a suitable video coding structure for wireless visual communications. Since the Robust Fine Granularity Scalability (RFGS) video coding technique could enhance the resistance against bandwidth variations and reduce the error drift effects due to data loss, we applied this technique over on H.264 and discuss the error resilience and error concealment problems over this H.264+RFGS structure.

In the following sections of this chapter, a brief introduction of H.264/AVC and RFGS is to be given. Some state-of-the-art tools of error resilience and error concealment will also be introduced.

2.2 Brief Introduction of H.264/AVC

H.264/AVC [1], the ITU-T Recommendation H.264 and ISO/IEC International

Standard 14496-10 (MPEG-4 part 10) Advanced Video Coding (AVC), is the newest video standard. The primary goals of H.264/AVC are to develop a video coding standard with extremely high coding efficiency and with a “network-friendly” video representation. It covers a Video Coding Layer (VCL), which efficiently represents the video content, and a Network Abstraction Layer (NAL), which formats the VCL representation of the video and provides header information in a manner appropriate for conveyance by particular transport layers or storage media. (Fig. 2.4)

As shown in Fig. 2.5, the scope of the H.264 standard includes only the central decoder, in which the bitstream syntax and the decoding process are described. This limitation of scope gives the maximal freedom to the encoder for all kinds of applications.

In the subsequent subsection, we’ll briefly introduce the H.264/AVC standard. Some key techniques in H.264/AVC, which improve the performance, are highlighted first. Then the network abstraction layer (NAL) and the video coding layer (VCL) are described individually.

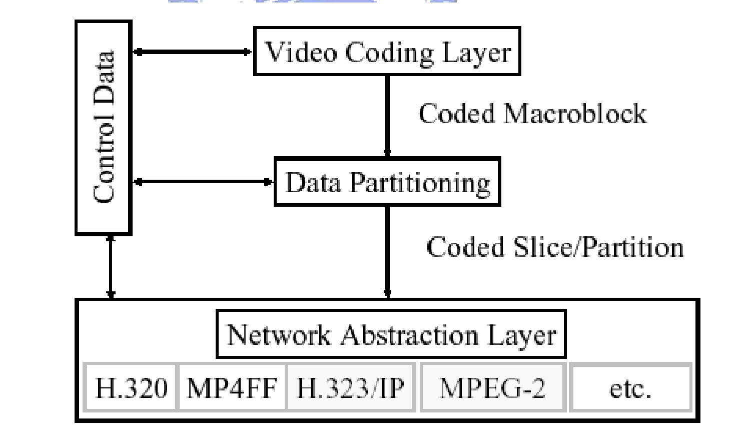


Fig. 2.4 Structure of H.264/AVC [2]

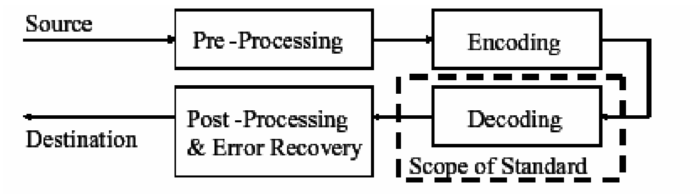


Fig. 2.5 Scope of H.264/AVC standardization[2]

2.2.1 Highlighted Features

Compared with prior video coding methods, several highlighted features in H.264/AVC enhance the coding efficiency. Some features improve the ability to predict the values of the picture content, and others directly improve the coding efficiency. It should be noted that there is no single feature that provides the majority of the significant improvement in coding efficiency. It is rather a plurality of smaller improvements that add up to the significant gain. On the other hand, in order to be robust to data errors/losses and be flexible for operations over a variety of network environments, there are features special for these purposes.

The following describes features in H.264 relevant to the ability of prediction [2]:

- Variable block-size motion compensation with small block sizes: the minimum luma motion compensation block size is 4x4.
- Quarter-sample-accurate motion compensation
- Motion vectors over picture boundaries
- Multiple reference picture motion compensation
- Decoupling of referencing order from display order
- Decoupling of picture representation methods from picture referencing capability
- Weighted prediction
- Improved “skipped” and “direct” motion inference
- Directional spatial prediction for intra coding
- In-the-loop de-blocking filtering

On the other hand, the features listed below directly improve the coding efficiency [2]:

- Small block size transform: a 4x4 transform. All major prior video coding standards use an 8x8 transform.
- Hierarchical block transform
- Short word-length transform: while previous designs have generally required

32-bit processing, H.264/AVC requires only 16-bit arithmetic.

- Exact-match inverse transform
- Arithmetic entropy coding
- Context-adaptive entropy coding

The features which help robustness to errors/losses and operation over a variety of network environments are listed in the following [2]:

- Parameter set structure: the transmission of some key information is separated to guarantee the conveyance of it.
- NAL unit syntax structure: a video packet format with greater customization when transmitting video data in a network.
- Flexible slice size
- Flexible macroblock ordering (FMO): a new ability to partition pictures into slice groups. Each slice group is independently-decodable.
- Arbitrary slice ordering (ASO)
- Redundant pictures
- Data partitioning: up to 3 partitions.
- SP/SI synchronization/switching pictures: specially-coded slices that enable efficient switching between video streams and efficient random access for video decoders.

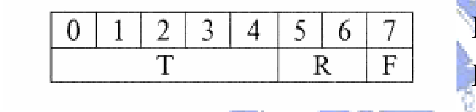
2.2.2 Network Abstraction Layer

The network abstraction layer (NAL) is designed to make the H.264/AVC code “network friendly”. It facilitates the H.264/AVC VCL (Video Coding Layer) data to transport layers. For example, in RTP/UDP/IP, there is exactly one NAL unit in one RTP packet for simple packetization [4]. Thus, packet losses in transport layers can be directly mapped to video packet losses in VCL.

2.2.2.1 NAL units

The coded video data is organized into NAL units, each of which is a video packet containing an integer number of bytes. The first byte of an NAL unit is a header byte, which contains the type information of the data in the NAL unit. The remaining bytes contain the coded symbols. The number of bytes in an NAL unit is variable.

Fig. 2.6 shows the format of the header byte. The NAL unit type (T) is a 5-bit field that characterizes the NALU (NAL unit) as one of 32 different types. The nal_reference_idc (R) indicates the importance of an NAL unit for the reconstruction process. ‘3’ means the most important while ‘0’ means the least. The forbidden_bit (F) is specified to be 0 in H.264 encoding.



0	1	2	3	4	5	6	7
T					R		F

T: NAL unit type
R: nal_reference_idc
F: forbidden_bit

Fig. 2.6 NALU header [4]

2.2.3 Video Coding Layer

The video coding layer of H.264/AVC, similar in spirit to other standards such as MPEG-2 Video, consists of a hybrid of temporal and spatial prediction, in conjunction with transform coding. Each coded picture is represented in block-shaped units called *macroblocks*. Fig. 2.7 shows the basic coding structure for H.264.AVC for a macroblock. A more detailed description of the technical contents of H.264 is given below [2]:

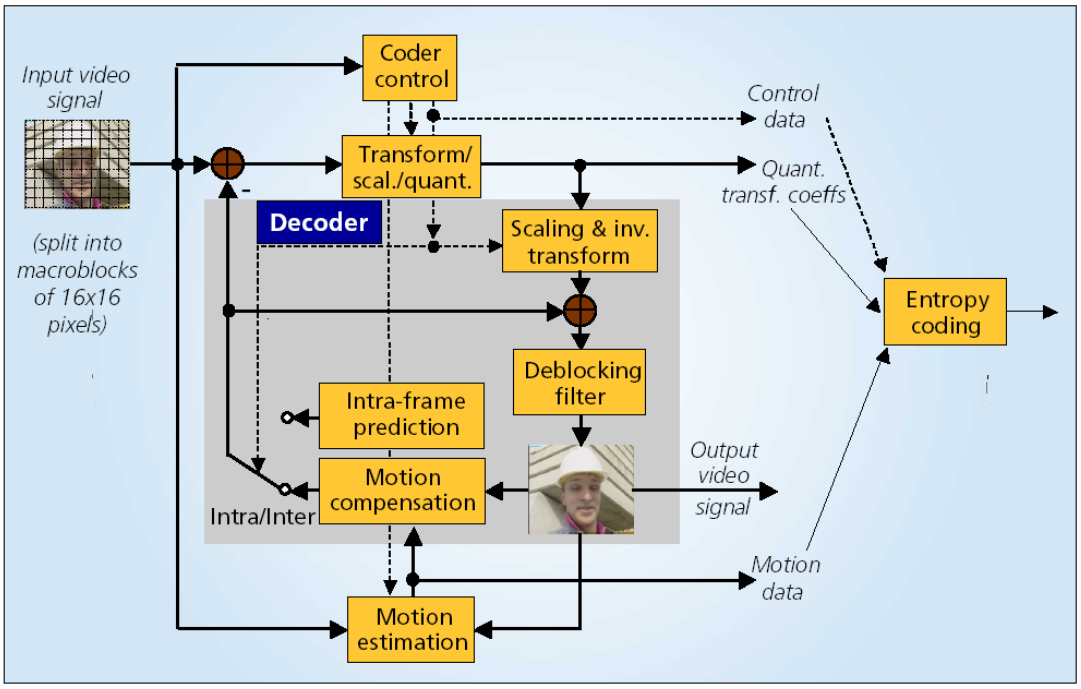


Fig. 2.7 Basic coding structure for H.264.AVC for a macroblock [5]

a. Pictures, frames, and fields

A coded video sequence in H.264/AVC consists of a sequence of coded pictures. A coded picture can be an entire frame or a single field. Furthermore, H.264/AVC supports macroblock-adaptive switching between frame and field coding. For that, a pair of vertically connected macroblocks is coded as two frame or field macroblocks.

b. YCbCr color space and 4:2:0 sampling

H.264/AVC is done by using YCbCr color space and 4:2:0 sampling with 8-bit precision per sample.

c. Slices and slice groups

Slices are a sequence of macroblocks which are processed in the order of raster scan. In H.264/AVC, the number of macroblocks in one slice is not constrained. See Fig. 2.8 as an example. In addition, flexible macroblock ordering (FMO) could be used in H.264/AVC. FMO partitions a picture into several *slice groups*. Each slice group is a set of macroblocks defined by a macroblock to slice group map and can be partitioned into one or more slices. Using FMO, a picture can be split into many macroblock-based scanning patterns, such as interleaved slices, a dispersed macroblock allocation, one or more “foreground” slice groups and a “leftover” slice group, or a checker-board type of mapping (See Fig. 2.9).

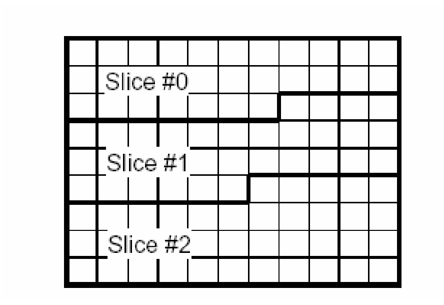


Fig. 2.8 Subdivision of a picture into slices [2]

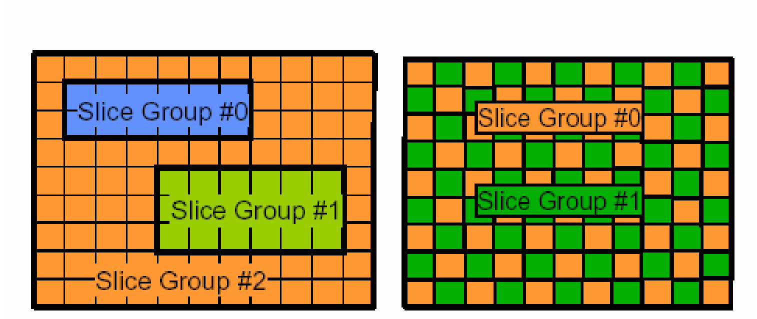


Fig. 2.9 Subdivision of a picture into slice groups using FMO [2]

No matter FMO is used or not, slices could be coded by different coding types as listed follows:

- I slice: coded by intra prediction
- P slice: coded by one-directional inter prediction
- B slice: coded by bi-directional inter prediction
- SP slice: switching P slices for efficient switching between different pre-coded pictures
- SI slice: switching I slice for random access and error recovery purpose.

d. Intra-frame prediction

There are several modes of intra coding in H.264/AVC, mainly Intra 4x4 prediction and Intra 16x16 prediction modes. In Intra 4x4 mode, the prediction is done for each 4x4 luma block. This mode is suitable for detailed regions of a picture. On the other hand, Intra 16x16 mode does prediction for the whole 16x16 luma macroblock and is mostly used on smooth areas of a picture.

When doing prediction, the neighboring samples of previously-coded blocks which are to the left and /or above the current block are used. In Intra 4x4 mode, the 16 samples of the 4x4 block are predicted by using one of nine prediction modes and the decoded samples in adjacent blocks. Fig. 2.10 shows five of the nine Intra 4x4 prediction modes. Intra 16x16 mode is similar to intra 4x4 mode but with only four prediction modes.

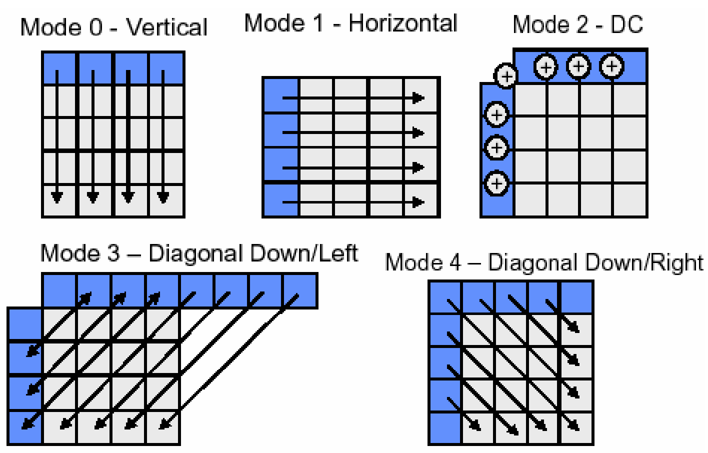


Fig. 2.10 Five of the nine Intra 4x4 prediction modes [2]

e. Inter-frame prediction

In H.264/AVC, various types of inter prediction are specified. When doing prediction, there are a number of block sizes to be used. They are 16×16, 8×16, 16×8, 8×8, 4×8, and 4×4, as shown in Fig. 2.11.

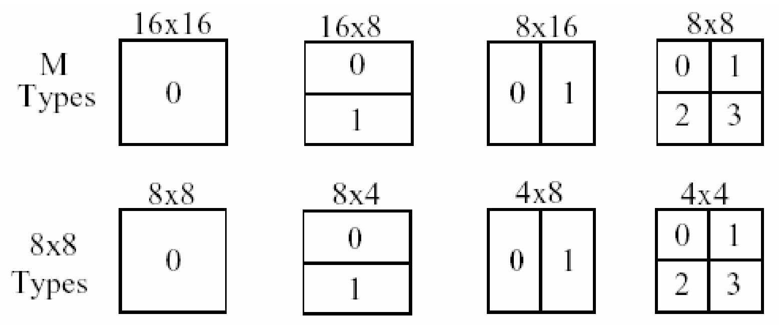


Fig. 2.11 Segmentations of a macroblock for motion compensation [2]
Top: segmentation of macroblocks
Bottom: segmentation of 8x8 partitions

The resolution of motion compensation is one quarter of the distances between two adjacent luma samples. The prediction values at half-sample positions are obtained by applying a one-dimensional 6-tap FIR filter horizontally and then vertically. At quarter-sample positions, the prediction values are generated by averaging samples at integer and half-sample positions.

Both d and e as mentioned above are performed in order to do a more accurate prediction so that the residual is reduced and thus enhance the coding efficiency.

f. Transform and quantization

H.264/AVC, as prior video coding standards, utilizes transform coding on the prediction residual. A 4×4 integer transform is used in H.264/AVC. The transform matrix is given as

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

Since an exact-match inverse transform exists, mismatches in inverse transform are avoided.

In addition, for each macroblock, a 4×4 Hadamard transform is applied to the DC coefficients of the 16 luma blocks, while a 2×2 Hadamard transform is applied to the DC coefficients of the 4×2 chroma blocks.

H.264/AVC uses the quantization parameter (QP: 1~52) to decide the quantization of transform coefficients. These values are arranged so that increasing 1 in QP means increasing the quantization step size by approximately 12%. It can be noticed that increasing the quantization step size by approximately 12% also means roughly reducing bit rate by approximately 12%.

g. Entropy coding

The arithmetic coding is applied in H.264/AVC. This coding scheme is proved to be more efficient than VLC coding. In addition, the concept of context-adaptation is adopted. It permits adaptation to non-stationary symbol statistics. The efficiency of entropy coding can be improved further if the Context-Adaptive Binary Arithmetic Coding (CABAC) is used.

h. In-loop de-blocking filter

H.264/AVC defines an in-loop de-blocking filter, which is applied in both the encoder and decoder. The strength of filtering is controlled by the values of several syntax elements, which are relevant to the difference between the two sides of an edge. With this filter, the blocking effect is reduced (See Fig. 2.12).

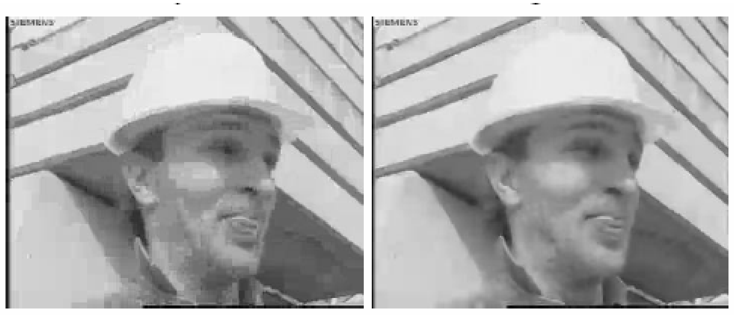


Fig. 2.12 Performance of the deblocking filter for highly compressed pictures [2]
Left: without deblocking filter, right: with deblocking filter

2.3 Scalable Coding

2.3.1 Scalability and FGS Coding

The goal of scalability coding is to optimize the video quality over a given bit rate range, instead of at a given bit rate. In Fig.2.13, the distortion-rate curve indicates the upper bound in quality for any coding technique at any given bit rate. Non-scalable coding tries to be as close to the distortion-rate curve as possible at a given bit rate; but when the channel bit rate is lower than the given rate, the received video quality becomes very poor. When the channel bit rate is higher than the given rate, nevertheless, the received video quality does not become any better. As to scalable coding, although its performance is not as good as non-scalable coding at each fixed bit rate, its curve may be parallel to the distortion-rate curve over a wide range of bit rates.

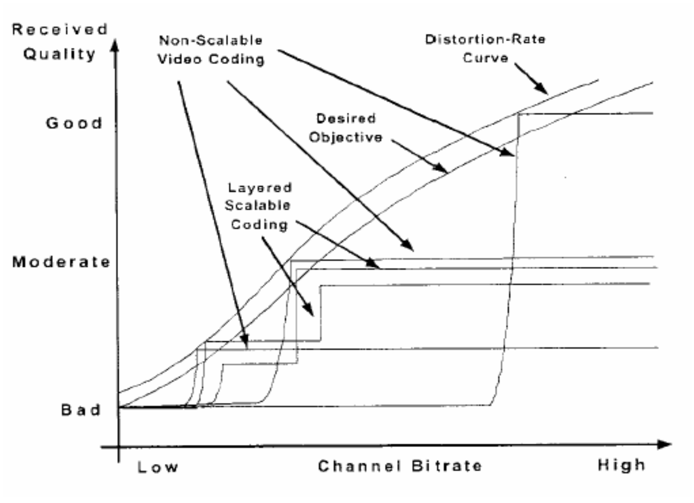


Fig. 2.13 Illustration of video coding performance [7]

Fig. 2.14 is the FGS (Fine Granularity Scalability) scalability structure [7]. FGS codes a video sequence into a base layer and an enhancement layer. The base layer uses non-scalable coding to achieve the lower bound of the bit-rate range. The enhancement layer is to code the difference between the original picture and the reconstructed picture. Fig. 2.15 shows the FGS encoder structure.

The enhancement layer is coded by bit-plane coding. For each 8x8 DCT block in the enhancement layer, a bit-plane of the block is defined as an array of 64 bits, taken one from each absolute value of the DCT coefficients at the same significant position. For each bit-plane of each block, (RUN, EOP) symbols are formed and variable-length coding is used to produce the output bitstream.

Besides the adaptability to bandwidth variation in Internet video streaming, FGS coding has some inherent error resilient properties. First, since the FGS enhancement-layer is intra-frame coded, there is no error propagation in the enhancement-layer. Next, as FGS is a scalable coding scheme, consisting of a base layer and an enhancement layer, different error-protection schemes could be employed for these two layers. Moreover, FGS base-layer has lower packet loss probability than single-layer. Therefore, FGS can be also regarded as a good error resilience tool.

The major disadvantage of FGS is the reducing of coding efficiency. This is because there is no inter prediction in the enhancement layer. RFGS (Robust Fine Granularity Scalability) coding is designed to be a flexible FGS framework that allows encoder to select a tradeoff between coding efficiency and video quality. RFGS may simultaneously maintain the coding efficiency with adequate video quality for varying bandwidths or error-prone environments. The detail of RFGS is to be briefly introduced in the following subsection.

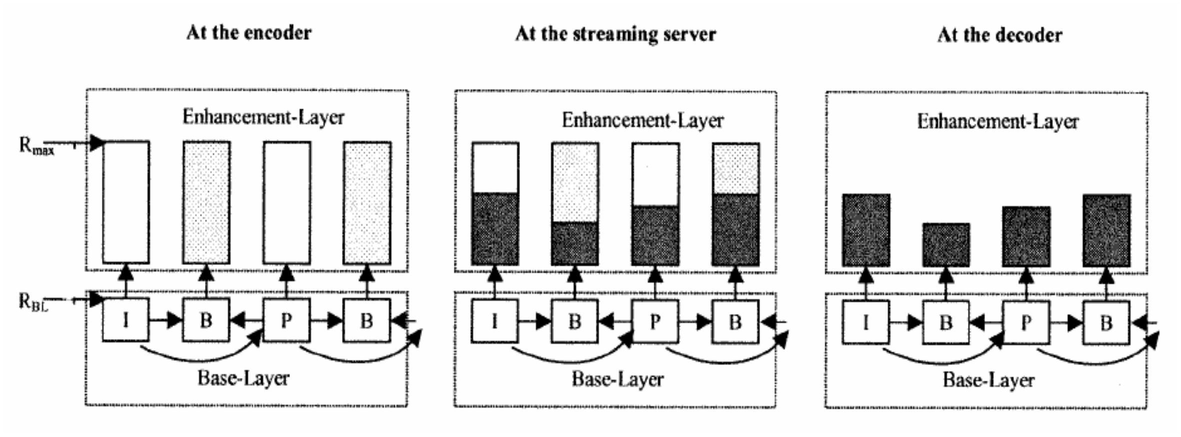


Fig. 2.14 FGS scalability structure [11]

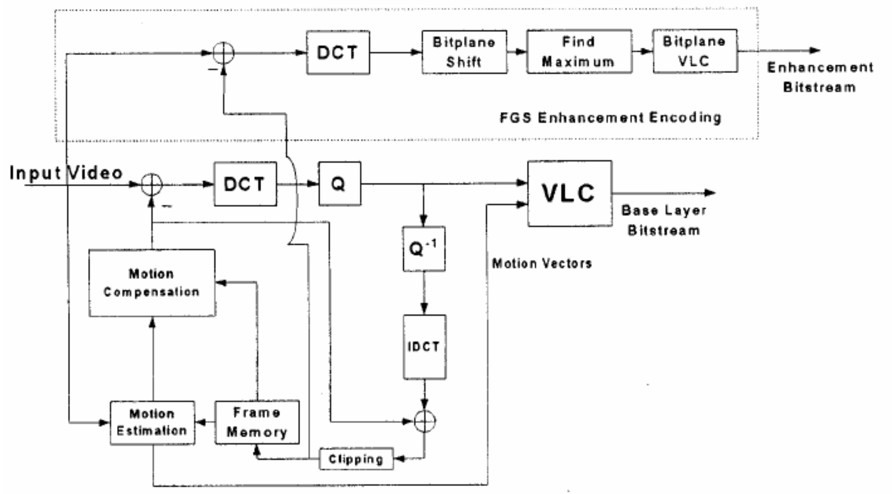


Fig. 2.15 FGS encoder structure [7]

2.3.2 RFGS Coding

Based on the structure of FGS, RFGS improves the coding efficiency by using ***leaky prediction*** and ***partial prediction*** on the enhancement layer [8].

As shown in Fig. 2.16, the enhancement layer has P bit planes. (H_{n1}~$H_{np,}$ n represent frame No.) Each frame has the flexibility to select the number of bit planes (denoted as β) used to form the high-quality reference frame for the prediction of the next frame's enhancement layer. For example, the first frame uses three bit planes to compute the high-quality reference frame. The above process is called ***partial prediction***. As β increases, the residuals decrease so that the coding efficiency is improved. However, once some enhancement data are not received at the decoder side, errors will increase and propagate to successive frames. Hence, β could be used to control the tradeoff between coding efficiency and error robustness.

Before doing motion compensation, the first β bit-planes will be scaled by a leak factor α, where $0 \le \alpha \le 1$. This process calls ***leaky prediction***. The leaky factor is used to speed up the decay of error energy in the temporal direction. Consequently, if any information at the first β bit planes is lost, the error will be α times attenuated for each frame at the enhancement layer. If $\alpha<1$, the error drift will be eliminated in a few frames. If $\alpha = 0$ or $\beta = 0$, the behavior of RFGS is just like the FGS coding scheme mentioned in the previous subsection.

In this thesis, we adopt RFGS coding as our coding structure for the following reasons:

1. It gives a mechanism to control the tradeoff between coding efficiency and error robustness.
2. Because RFGS is developed by the research team of Prof. Chiang at National Chiao Tung University, it is relatively easy to get the source code.

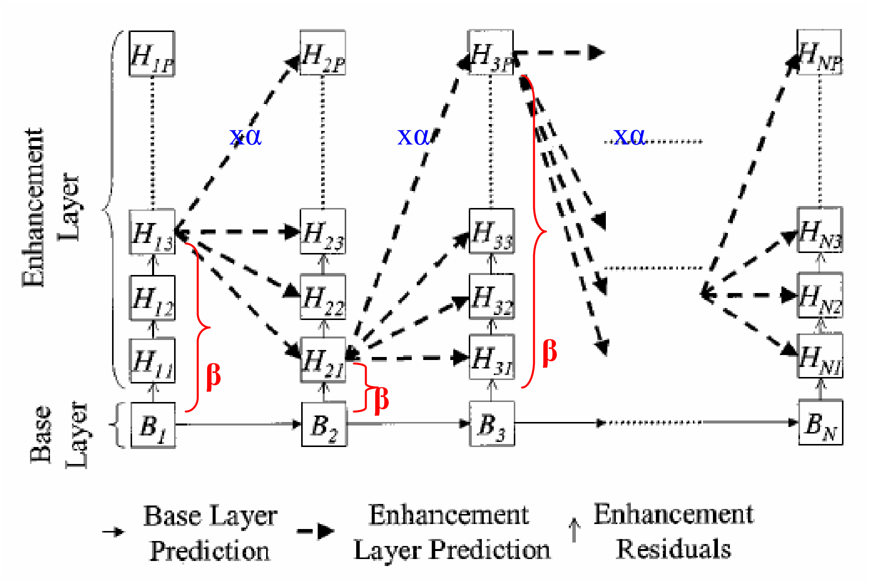


Fig. 2.16 Diagram of RFGS partial inter-prediction mode [8]

2.3.3 H.264 + RFGS Coding

H.264 is a non-scalable coding standard, and RFGS is a tool of scalability. Applying RFGS on H.264 generates a scalable coding. Similar to the structure mentioned in Section 2.2.2. A video sequence is partitioned into the base layer and the enhancement layer in H.264+RFGS coding. The base layer uses non-scalable coding, i.e. H.264 video coding. The enhancement layer is coded by bit-plane coding (See Fig. 2.17). The coding efficiency of H.264+RFGS is lower than H.264, because non-scalable coding may focus on the optimization of a specific bandwidth while the scalable coding must consider over a range of bit rate. Sacrificing coding efficiency, both the adaptation of bandwidth variation and the robustness of error resistance are improved. Further discussion is given in Chap 3.

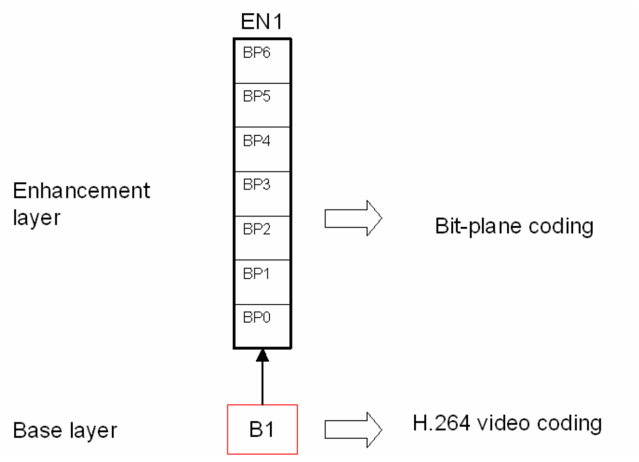


Fig. 2.17 Hybrid coding scheme of H.264+RFGS
BPn: the nth bit-plane (Assume there are 6 bit-planes.)

2.4 Error Control Techniques

The goals of error control are (1) to fix the errors, (2) to limit the spatiotemporal error propagation, and (3) to minimize the visual effect of errors within one frame. We often attempt to achieve the first two goals at the encoder side by using techniques of error resilience, and try to achieve the third one at the decoder side by using tools of error concealment. As described at Section 2.3, we applied RFGS coding on H.264 as an error resilience tool to limit the spatiotemporal error propagation. Based on the H.264+RFGS structure, we study for more suitable methods to improve the error robustness of H.264. However, while the ability of error robustness is promoted, we must avoid a huge increase of overhead. Moreover, considering the necessity of real-time applications, the tools of error resilience should not be too complex.

The following subsections introduce some error control techniques which have been brought up in recent publications.

2.4.1 Tools of Error Resilience

2.4.1.1 Error Resilient Scalable Video Coding

A scalability structure could be a good error resilience tool because of the following characteristics:

(1) Reduced error propagation

In RFGS, the use of α may reduce the effect of error propagation.

(2) Lower packet loss probability for the base-layer than for single-layer coding

Since the amount of data to be protected is lower for the base-layer of a scalable codec than for a non-scalable (NS) codec operating at the same bit-rate, for an identical packet loss rate, the number of packets lost from the base-layer is lower than that for the NS stream. This means that in a scalable coding scheme the significant part of the video data has a smaller probability to get lost. Therefore, for an identical packet loss rate, a scalable coding scheme tends to perform better than the NS coding scheme at the same bit rate.

The aforementioned characteristics are based on the inherent properties of scalable coding. Further, scalable coding can be more error resilient by using unequal packet-loss protection (UPP) [10~12]. UPP could help allocate system resources more efficiently and could provide a good quality of service by applying higher level protection on more important data. Since for a scalable video coding, the base layer is more important than the enhancement layer, a higher protection level could be applied to the base layer. This unequal protection makes scalable video coding more error resilient.

To provide UPP, several different techniques could be employed. If we only operate in the application layer (the 7th layer in the OSI model), unequal FEC is commonly used. On the other hand, if channel coding is also used, techniques of cross layer FEC, retransmission, etc., could be applied. In the following paragraphs, we introduce these two types of strategies.

(1) In application layer [11]

In application layer, we use unequal FEC to implement UPP. Here FEC indicates the FEC across packets. As shown in Fig. 2.18, the video bitstream is written line-wise into q-packets. Additionally, m-q packets are generated by forming codewords column-wise.

For the realization of the FEC, RS codes have been widely used due to their flexibility. An (m, q) RS code realizes the structure in Fig. 2.18, and it can reconstruct the codewords entirely as long as the number of packet losses is less than m-q packets. There are quite a few different levels of RS codes which have different values of m and q . Using these different levels of RS codes, we can realize the UPP.

For example, Fig. 2.19 shows the 3-bit planes of FGS, a widely used structure of scalable video coding. According to the importance, we apply Bit-plane 1 the largest ratio of redundancy while Bit-plane 3 the least.

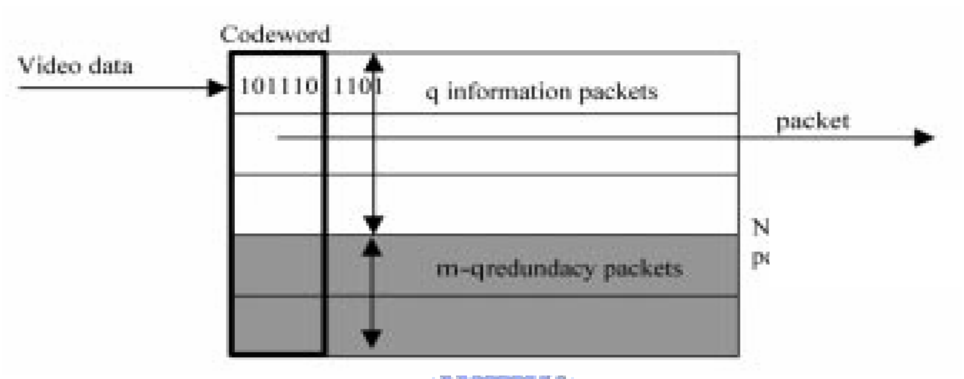


Fig. 2.18 FEC across packets [11]

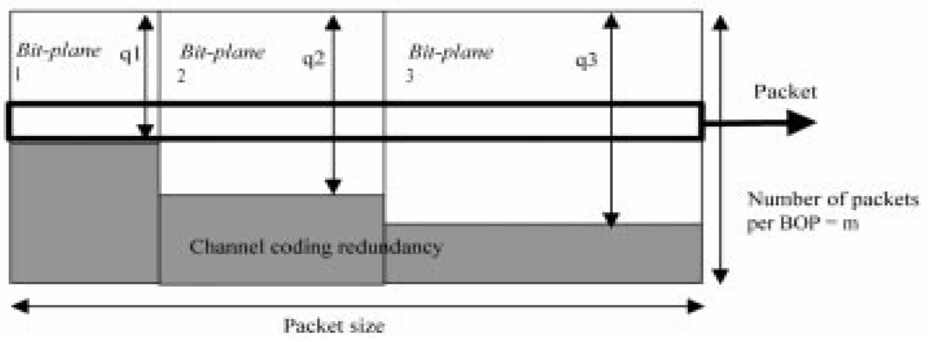


Fig. 2.19 Transmission of an FGS stream containing n = 3 bit-planes [11]

(2) Combined with channel coding (source-channel coding)

In this type, we found two main methods in current researches.

1. Cross layer FEC [12]

As shown in Fig. 2.20, in physical layer, each packet is protected by RS codes. More significant data are allocated more parity bits. Besides, in application layer, video packets of more significant data are protected by adding cross-packet RS codes.

2. Adaptive application-layer FEC and physical layer automatic repeat request (ARQ) [10]

ARQ is a mechanism which controls the retransmission of a loss packet. As shown in Fig. 2.21, the PC sends a data frame and then expects a CF-ACK frame from the corresponding station within a short inter-frame space (SIFS) time. After receiving the CF-ACK frame, the PC sends the next data frame. If the data frame is received in error or if the frame is not received at all, the PC starts the re-transmission of the data frame after a priority inter-frame space (PIFS) time. If the data is received correctly but the CF-ACK frame fails, then the PC waits for a SIFS time and re-transmits the data frame. The number of re-transmission is limited, denoted as R. Combining ARQ and the application-layer FEC, we can look for an adaptive method to get a more error robust source-channel coding. According to the research done in [10], at poor channel condition, using application layer FEC and a larger R gives better results; otherwise, at good channel condition, using no FEC and a larger R has the best results.

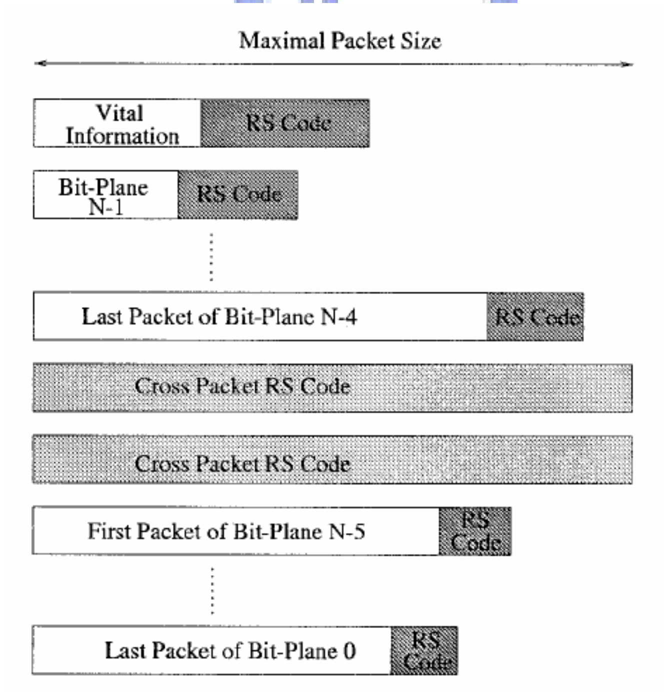


Fig. 2.20 Bit-stream packetization with cross layer UPP [12]

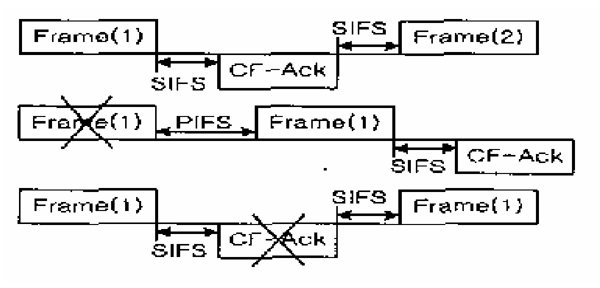


Fig. 2.21 ARQ mechanism [10]

2.4.1.2 Error Resilient Tools in H.264

H.264 itself also includes a number of error-resilience tools. Here we give a brief introduction [4].

a. Intra placement

Intra placement is used primarily to combat drifting effects; i.e. to reduce the error propagation. H.264 allows macroblocks (MBs) as Intra placement unit, instead of a slice or a picture. It means we can put an intra MB at any position of any picture. This makes intra placement more flexible so that the bit-rate peak of an intra picture can be avoided.

b. Picture segmentation

No constraints are imposed on the number of MBs in a slice in H.264. Therefore we can packetize video coding streams into different sizes of video packets. (In a simple structure, a slice is packetized into a packet.) There is a close relation between packet size and packet loss probability at an identical given condition. As packet size gets smaller, packet loss probability becomes lower.

c. Reference picture selection

H.264 allows at most five reference pictures. This is a useful error resilience tool in feedback-based systems. In a feedback-based system, the encoder receives information from the decoder about the information of lost or damaged picture areas. The encoder can react to the received information by choosing a correctly received reference frame. For a system without feedback, the factor of more than one reference pictures has no apparent benefits.

d. Data partitioning

In H.264, three different partition types are used:

1. Header information, including MB types, quantization parameters, and motion vectors, is the most important information.
2. Intra partition
3. Inter partition

Data partitioning can help us to do higher level protection on more important data, like the header information. With this arrangement, when the Inter or the Intra partitions are missing, a comparatively high reproduction quality can be achieved as if only the texture information is missed. This is due to the availability of MB types and the motion vectors in the header information,

e. Parameter sets

There are two parameter sets in H.264. They are sequence parameter set and picture parameter set. The sequence parameter set contains all the information related to a sequence of pictures, and a picture parameter set contains all the information related to the slices belonging to a single picture. They both can be packetized into a packet and be sent to the receiver in a more reliable way. This guarantees the arrival of the vital important information at the receiver.

f. Flexible macroblock ordering (FMO)

H.264 allows us to assign MBs to slices in an order other than the scan order. This ordering is called FMO. To do so, each MB is statically assigned to a slice group using an MB allocation map (MBA map). Within a slice group, MB's are coded using the normal scan order and can form one or more slices. For example, in Fig. 2.22, all MB's are allocated either in slice group 0 or slice group 1 in a checker-board mode. Obviously, when losing one of the two slice groups, each lost MB has four neighboring MB's to be used to conceal the lost information.

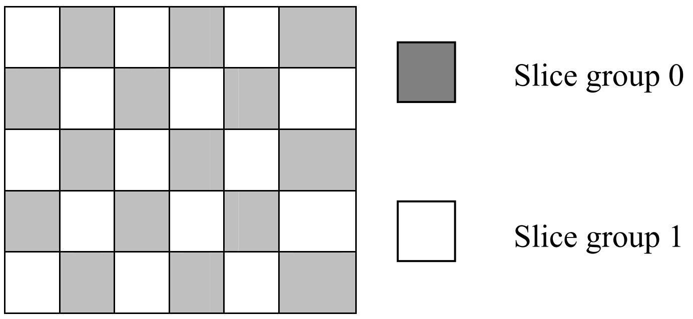


Fig. 2.22 Two slice groups of checker-board FMO

g. Redundant Slices(RS's)

H.264 allows an encoder to place one or more redundant representations of the same MB's into the same bitstream, but packetized into different packets. These redundant representations are called redundant slices (RS's). For example, the encoder can code low QP representations as RS's. Therefore, once the primary data are lost, the RS's can be used to reconstruct the video of lower but acceptable performance.

2.4.2 Tools of Error Concealment

At the decoder, we can use some post-processing methods to conceal the errors in a reconstructed video. Many post-processing error concealment methods have been proposed. They can be divided into three categories: frequency, spatial, and temporal. There are also hybrids of these three. Moreover, these methods can be made to be adaptive [13].

a. Frequency concealment

Working in the frequency domain, DCT coefficients of missing blocks are estimated using the corresponding DCT coefficient of neighboring blocks, or using the neighbors' DC values, or other neighborhood features. These methods usually attempt to estimate low frequency coefficients.

b. Spatial concealment

Working in the spatial domain, one can do interpolation directly using neighboring blocks. We know that the correlation between lost pixels and the pixels used to do the interpolation will be larger when these pixels get closer. So it has been proposed in [9] to interpolate pixel values within a damaged MB from its four 1-pixel-wide boundaries. A pixel may be interpolated from two nearest boundaries, as shown in Fig. 2.23(a), or from all four boundaries, as shown in Fig. 2.23(b).

c. Temporal concealment

In temporal concealment, blocks from other frames are used for concealment, either by attempting to reconstruct the motion vector of the lost MB, or by searching for a block that has a good match to the sides and neighborhood of the missing block. If the estimation of a motion vector is inaccurate, the block obtained will have artifacts at the boundaries. The lost MV could be estimated by using the average or median of the neighboring MV's from the above, below, and diagonal directions or by using the motion vector of the neighboring block which produces the best match for the boundary pixels.

Here gives a new temporal concealment method proposed by T. Shanableh and M Ghanbari [14]. Owing to the bidirectional prediction of B-pictures, the motion vectors of both past and future P-pictures (or I-pictures) can be related to those motion vectors of the B-pictures in the same subgroup. Fig. 2.24 shows this relation for sequences. If losses occur in I- and P-pictures, motion vectors could be derived from the neighboring B-pictures. For example, as shown in Fig. 2.24, if there are lost motion vectors in P_7-picture ($V_{4\to7}^{fwd}$), these vectors could be estimated from the motion vectors of B_8- and B_9-pictures:

$$\textit{estimate}\ (V_{4\to7}^{fwd}) = V_{4\to9}^{fwd} - V_{7\to9}^{bwd}$$

or

$$\textit{estimate}\ (V_{4\to7}^{fwd}) = V_{4\to8}^{fwd} - V_{7\to8}^{bwd}$$

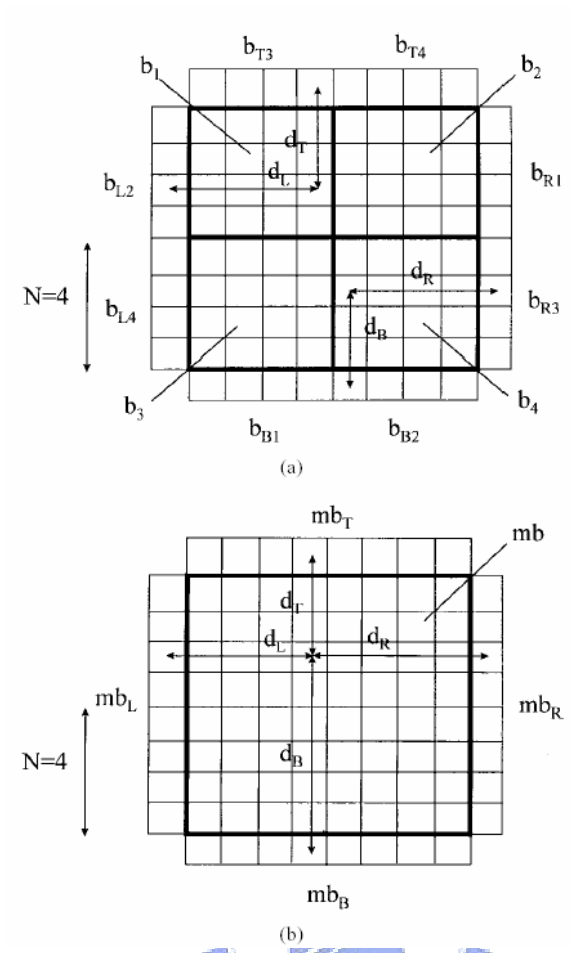
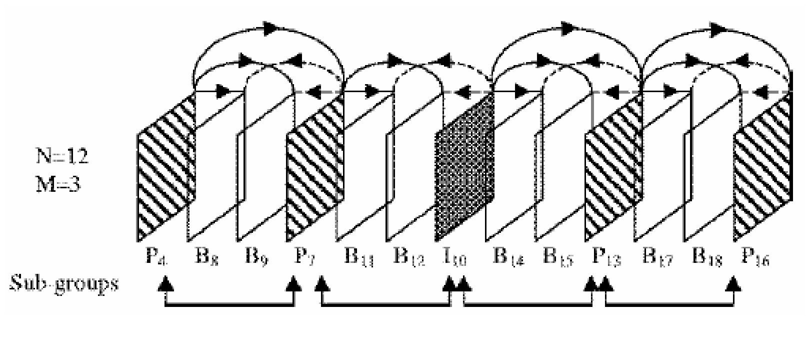


Fig. 2.23 Spatial interpolation for error concealment [9]

(a) Block based. (b) MB based.



N=12
M=3
P_4 B_8 B_9 P_7 B_{11} B_{12} I_{10} B_{14} B_{15} P_{13} B_{17} B_{18} P_{16}
Sub-groups

Fig. 2.24 Direction of the prediction motion vectors in an MPEG sequence [14]

d. Hybrid methods

A variety of hybrid algorithms, which combine more than one of the frequency, spatial, and temporal approaches, have been proposed in the literature. For example, in temporal concealment, the quality of the referenced block can be improved by spatial smoothing at its edges to make the block ~~it~~ conform to the neighbors [9].

Here gives another example which combines both frequency and spatial concealments. Fig. 2.25 illustrates two methods to estimate missing DCT coefficients by making use of spatial smoothness. When only the DC coefficient of an MB is lost, the DC coefficient value can be estimated by considering the smoothness of the boundaries of that MB. However, when several additional low-frequency coefficients are also lost, the smoothness in the MB should be put into consideration too.

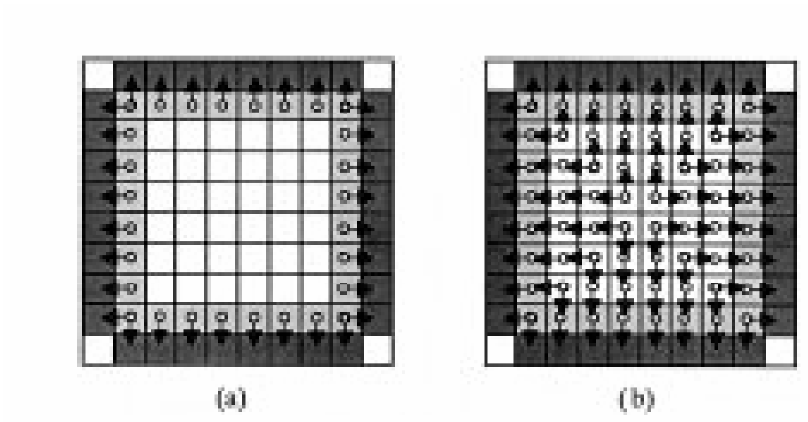


Fig. 2.25 Estimation of DCT coefficients of missing blocks based on ~~by~~ neighboring blocks [9]
(a) When only the DC coefficient is lost.
(b) When the DC and several low-frequency coefficients are missing.

e. Adaptive methods

It is proposed that a few error concealment methods can be switched adaptively according to some conditions, like the amount of loss data, the characteristics of video content, etc. To make methods adaptive, designing decision trees could be a good choice. Decision trees can help choosing a suitable method among several ones by examining some characteristics. What characteristics in a video should be considered could be set manually.

Fig. 2.26 shows an example of decision trees [13]. In this decision tree, there are five terminal-nodes. The ovals show the test which is used to split the data in that node. The percentages show the amount of data in each node that has the spatial, panning, or copyPmb concealment method as the best concealment method.

Methods for concealment –
spatial: interpolate linearly from boundary pixels in top/bottom MBs
panning: use the camera panning motion vector
copyPmb: copy co-sited MB from the previous P frame if it's Intra-coded or has MB=0

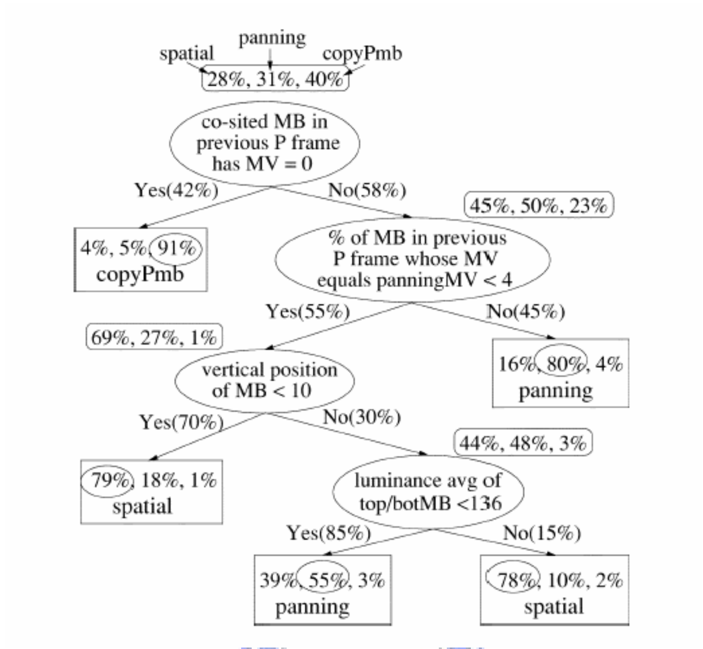


Fig. 2.26 An example of the decision tree for error concealment [13]