Module

8

VIDEO CODING STANDARDS

Lesson

27

H.264 standard

Lesson Objectives

At the end of this lesson, the students should be able to:

1. State the broad objectives of the H.264 standard.
2. List the improved prediction methods adopted in H.264
3. Implement motion estimation with quarter-pixel accuracy for video sequence
4. Present the concept of multi frame motion compensation
5. Explain the principle of deblocking filter
6. Illustrate the intra-frame prediction modes of H.264
7. List the improved transform and entropy coding schemes
8. Implement 4 x 4 integer transforms
9. Explain the basic concepts of two entropy coding schems- CAVLC and CABAC
10. List the features of the Video Coding Layer (VCL)
11. Define 'slice and 'slice group'
12. Explain the concept of Flexible Macroblock Ordering (FMO)
13. State the role of the network adaptation layer (NAL)

27.0 Introduction

The video coding standards released by the International Telecommunication Union (ITU) in the 1990s, viz H.261 and H.263 (lesson-26) along with its extensions H.263 + triggered wide range of applications, which did not remain restricted to ISDN and Public Switched Telephone Networks (PSTN) domain, but proliferated to mobile wireless networks, LAN/internet delivery of video stream etc.

The need for further improvement in coding efficiency by at least two times for the same fidelity was soon realized. In 1998, the Video Coding Experts Group (VCEG) of the ITU invited proposals for a new video coding project, named H.26L which would have two times better coding efficiency over a broad range of applications. In December 2001, the VCEG and the Motion Pictures Experts Group (MPEG) formed a Joint Video Team (JVT). Their combined efforts resulted in the new coding standard H.264. This also forms the Part-10 (Advanced Video Coding) of MPEG-4 and is therefore referred to as H.264 / AVC standard.

Some of the major highlighting features of this latest video coding standard are improved motion estimation up to quarter-pixel accuracy; use of 4 x 4 integer transforms in place of 8 x 8 DCT; improved context based arithmetic entropy coding; advanced prediction modes for intra and inter-coded frames etc. In this lesson, we are going to cover this standard with some details.

27.1 Broad objectives of the H.264 standard

The H.264 standard was designed for enhanced compression performance with network friendly features to address a broad range of applications that include conversational (e.g. Video telephony and Video conferencing) and non conversational (e.g. storage, broadcast and streaming) applications. Its application domain may be broadly classified as

- Broadcast over cable, modems, ADSL, terrestrial etc.
- Interactive storage such as optical and magnetic devices, DVDs etc.
- Conversational service over ISDN, Ethernet, LAN, modems, DSL wireless and mobile networks etc, or their combinations.
- Video-on-demand or streaming service over all the above networks.
- Multimedia messaging service (MMS) over all the above networks.

27.2 Improved Prediction modes in H.264

The H.264 standard could achieve a major breakthrough in coding efficiency through several improved prediction mechanisms. Some of the major ones are listed below:

(a) *Variable block size motion compensation*: Block size for motion vectors may be as small as 4 x 4.

(b) *Quarter-sample accurate motion compensation*: This results in much improved prediction performance, as compared to half-pixel accurate motion estimation in H.263. The interpolation scheme for quarter-pixel motion estimation will be explained later.

(c) *Motion vectors over picture boundaries* – This is an optional feature in H.263, but incorporated in H.264.

(d) *Multiple references picture motion compensation* – In previous standards, only the immediate past frame can be used as the reference for motion estimation. In H.264, the encoder can select among a larger number of stored and decoded past frames.

(e) *Directional spatial prediction for intra coding* – This feature significantly improves the intra-coding performance and is explained in details later.

(f) *In the loop deblocking filter* – Blocking artifact is a serious problem in block based very low bit rate video coding. A deblocking filter, implemented within the motion compensation loop of the encoder reduces blocking artifacts. This is explained later.

27.2.1 Motion estimation with quarter pixel accuracy:

In H.264 standard, the accuracy of motion estimation is in unit of one quarter of the distance between the luminance samples. In case the motion vector points to integer sample positions, the predicted signal can be obtained directly from the reference frame. For fractional values of motion vector components, the predicted signal is obtained through interpolation to generate the fractional positions.

The fractional pixel interpolation scheme is illustrated in fig 27.1.

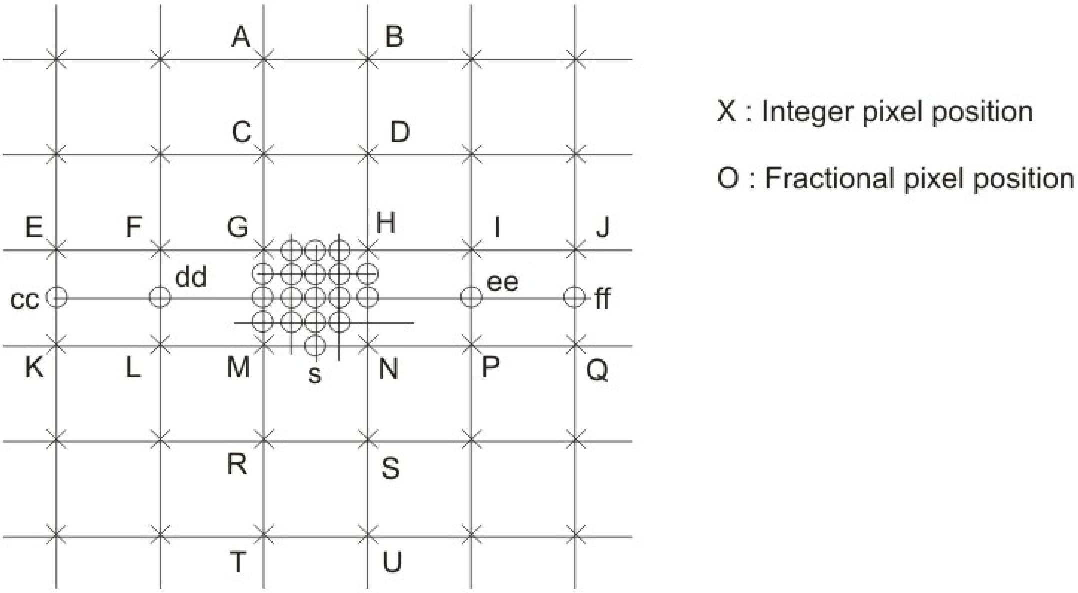


Fig. 27.1 Fractional pixel interpolation.

The pixels at the integer positions are indicated by "X" symbols and those at fractional pixel positions are indicated by "O" symbols. The pixels at half-sample positions are interpolated by applying one dimensional 6-tap FIR filter horizontally and vertically. For examples, the samples labeled 'b' and "h" are derived by first calculating the intermediate values b_1 and d_1 through 6-tap filter as

$$b_1 = E - 5F + 20G + 20H - 5I + J$$

$$h_1 = A - 5C + 20G + 20M - 5R + T$$

The final prediction values of b and h are obtained as

$$b = (b_1 + 16) >> 5$$
$$h = (h_1 + 16) >> 5$$

Left shift by 5 – bits (i.e. division by 32) restricts the *b* and *h* values in the range of 0 to 255.

The intermediate interpolated value at position "*j*" is given by

$$j_1 = cc - 5\,dd + 20\,h_1 + 20\,m_1 - 5\,ee + ff$$

The final prediction value is given by $(j_1 + 512) >> 10$ to restrict the value in the range of 0 to 255.

The pixels at quarter-sample positions, labeled *a, c, d, n, f, i, k* and *q* are derived by averaging with the pixels at nearest integer/ half sample positions, with upward rounding. For example, the pixel at "*a*" is derived as

$$a = (G + b + 1) >> 1$$

The pixels at quarter-sample positions, labeled *e, g p* and *r* are derived by averaging with the pixels at nearest integer/ half- samples positions in the diagonal directions, with upward rounding. For examples the pixel at '*e*' is derived as

$$e = (b + h + 1) >> 1$$

The chrominance component predictions are done by bilinear interpolation.

27.2.2 Multi-frame motion compensation :

The H.264 standard supports multi frame motion compensation, in which more than one prior coded frames may be used as reference.

This scheme requires storage of a few references pictures at both encoder and the decoder. The concept of multiple reference frames is illustrated in fig 27.2 for four prior decoded frames Fig.27.2

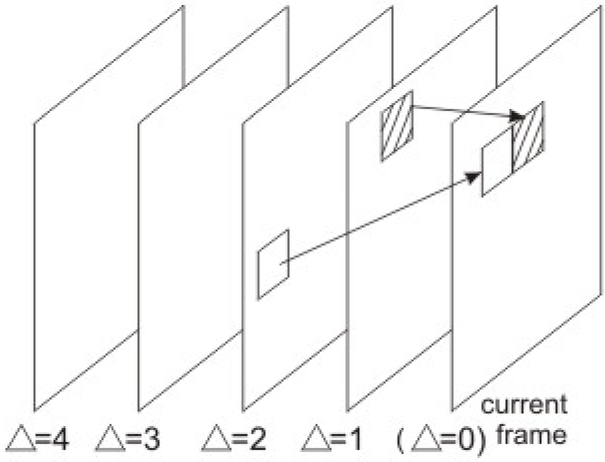


Fig. 27.2 Multiple reference based prediction.

When the size of a multiple reference buffer is set to one, the prediction mechanism is same as that in H.261 and H.263. For the H.264, it is possible to specify different reference frame numbers (called as index parameters) for different luminance blocks of size 16 x 16, 16 x 8, 8 x 16 or 8 x 8. Motion compensation on blocks smaller than 8 x 8 (i.e., 8 x 4, 4 x 8 or 4 x 4) are required to be done on the same reference index for all blocks within the 8 x 8 region.

27.2.3 In loop deblocking filter :

H.264 defines as adaptive in loop deblocking filter, where the strength of the filtering is adaptively controlled to reduce the effects of blocking artifacts. The deblocking filter is brought within the motion compensated prediction loop.

The basic philosophy of in loop deblocking filter is that if the absolute difference in intensity of the two adjacent pixels at the edges of two neighbouring block is relatively large, it is considered to be resulting out of blocking artifact and should be filtered. However, if the absolute difference is so large that it is unlikely to result from the coarseness of quantization, it should represent a genuine intensity transition and should not be smoothened.

The extent of filtering depends upon the quantization parameters and is larger for coarser quantization. Deblocking filter typically reduces the bitrate by 5% -10%, while producing the same objective quality as the unfiltered video.

27.2.4 Intra-frame prediction modes in H.264:

In H.264, each macroblock can be encoded in one of the several coding types. The types include *intra coding*, which we had already explained in the previous standards. H.264 supports following three types of intra-coding –

- *Intra – 4 x 4*

- *Intra – 16 x 16*
- *I-PCM*

The *Intra- 4x4* type predicts each 4x4 luminance block separately, based on its previously coded neighbouring blocks, which are either to the left and/or above the block to be predicted. This coding type is preferred for those regions of a picture which are required to be encoded with significant detail.

The *Intra –16 x 16* type on the other hand predicts the entire macroblock, based on the previously coded neighbouring macroblock and is preferred for smooth areas of a picture.

The third coding type, *I-PCM* allows the encoder to bypass the prediction and transform coding processes and instead directly send the values of the encoded samples.

The Intra 4 x 4 type has nine prediction modes of which eight are based on directions and one is the dc prediction mode (mode-2) which uses same prediction values for all the 16 pixels. The eight prediction directions are shown in fig 27.3 with the mode numbers.

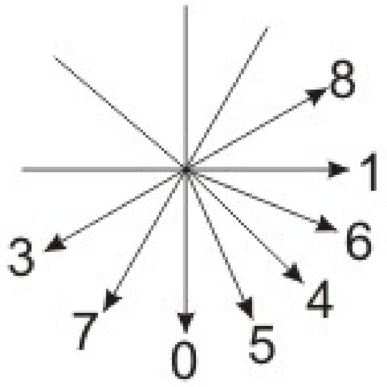
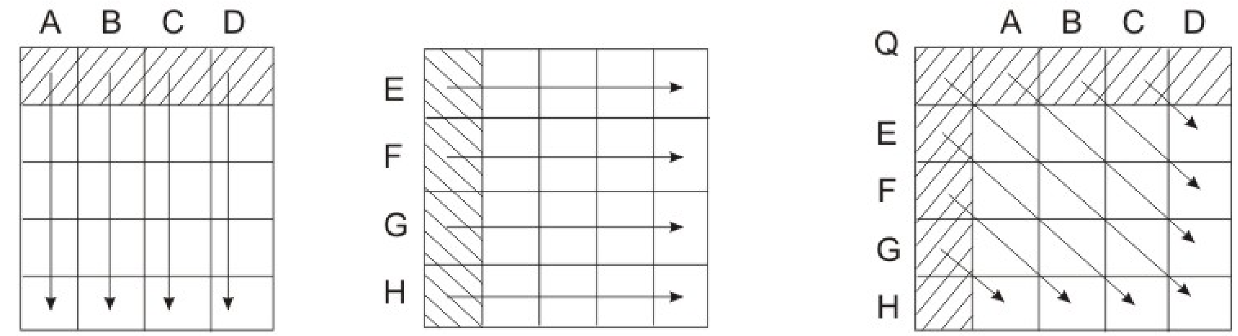


Fig. 27.3 Intra prediction directions.



(a) Mode - 0 (vertical) (b) Mode - 1 (horizontal) (c) Mode - 4 (diagonal down/right)

fig 27.4 Examples of Intra -4x 4 prediction modes

Examples of three predictions modes (mode-0, mode – 1 and mode-4) are illustrated in fig.27.4. As shown in fig.27.4(a), the pixels A, B, C and D lying just above the 4 x 4 block being predicted are copied vertically down along the columns. Fig 27.4(b) shows similar predictions horizontally in mode-1, where the four pixels E,F.G and H lying just to the left of the 4 x 4 block are copied horizontally along the rows. Fig 27.4(c) illustrates the diagonal prediction in *mode-4*.

The *Intra-16 x16* type supports only four predictions modes –

- ***Mode – 0*** **:** Vertical prediction
- ***Mode – 1*** ***:*** Horizontal prediction
- ***Mode – 2*** ***:*** DC prediction
- ***Mode – 3*** ***:*** Plane prediction, details of which is explained in the specifications.

27.3 Improved transform coding and entropy coding schemes

Apart from improved prediction methods explained in the previous section (section 27.2), the H.264 standard offers significant performance improvements through its better transform coding and entropy coding mechanism. Some of the major improvements are listed below :

- ***Smaller block-size integer transforms*****:** H. 264 uses a 4 x 4 transform based on integer coefficients instead of 8 x 8 block size DCTs used in the earlier standards. Details of the integer transform are explained in the next subsection.

- ***Improved entropy coding scheme:*** Two entropy coding schemes are supported in the H.264 standard. One is called the Context Adaptive Variable Length Coding (CAVLC) and the other is the Context Adaptive Binary Arithmetic Coding (CABAC). These two schemes are discussed in section 27.3.2

27.3.1 Transform Coding in H.264

Like its predecessors, H.264 also uses transform coding techniques to encode the prediction error residual. However, unlike its predecessors, this standard does not use DCT over 8 x 8 blocks. Instead, it uses a separable integer transform with properties similar to a 4 x 4 DCT.

The elements of the 4 x 4 integer transform matrix is given below :

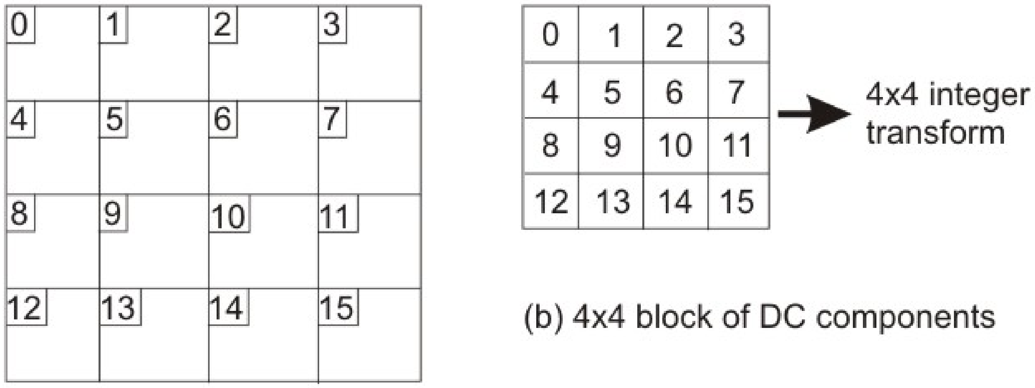
$$H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

The inverse transform is also defined by exact integer operations and hence errors arising out of finite arithmetic precision are avoided.

Smaller size transforms results in following advantages :

(a) blocking artifacts are reduced
(b) involves less computations- since the transform matrix elements are +1, -1, +2 and –2, only add/ subtract and shifts are required
(c) improved prediction process for inter and intra modes and has less spatial correlation.

In most cases, using the small size transform is perceptually better but in some pictures, which have smooth intensity regions, a larger size transform would be preferred. The H.264 standard achieved this by applying a repeated transform, as illustrated in fig 27.5.



(b) 4x4 block of DC components

(a) 16x16 macro block

Fig. 27.5 Repeated transform applied on a 16x16 macro block.

Fig 27.5 shows a macro block of 16 x 16 pixels, subdivided into 16 numbers of 4 x 4 blocks, each of which are integer transformed. The dc components for each of the blocks are indicated with the block number index. The 16 DC coefficients are again arranged as a 4 x 4 matrix, as shown in fig.27.5(b) and the 4 x 4 integer transform is re-applied. The transform coefficients now cover the entire macroblock in very smooth situations. An additional 2 x 2 transform is also applied to the DC coefficients of the four 4 x 4 blocks of each chroma component.

27.3.2. Entropy coding in H.264:

The H.264 standard supports two entropy coding methods:

(a) Context Adaptive Variable Length Coding (CAVLC)

(b) Context Adaptive Binary Adaptive Coding (CABAC)

In the CAVLC Scheme, instead of using a single VLC table for all syntax elements, different VLC tables are designed for different syntax elements and the tables are switched depending upon the already transmitted syntax elements.

In this coding scheme, number of non zero quantized coefficients (N) in a block and the actual value as well as the position of the transform coefficients are encoded separately.

To convey the information of quantized transform coefficients for a 4 x 4 luminance block, following data elements are used –

- *Number of nonzero coefficients (N)* and trailing 1s at the end of the zigzag scan order.

- *Coefficient values*- These are scanned in the reverse scan order, since the spread of coefficient values is less for the higher frequency components than for the lower frequency components than for the lower frequency ones. Encoding this may involve VLC table sartching.

- *Coefficient sign*.

- *Total Zeroes*: The number of zeros between the last non zero coefficient of the scan and its start.

- *Run before*: It specified the position of the zeros.

The CABAC scheme improves the coding efficiency further. The use of arithmetic coding permits assignment of non integer number of bits to each symbol of the alphabet. Furthermore, CABAC uses context modeling where the statistics of already coded syntax elements determine the conditional probabilities of the symbols based on which the arithmetic codes are decided.

27.4 Features of H.264 Video Coding Layer (VCL)

The H.264 encoder consists of two different data layers. The encoded video data, which includes the entropy encoded integer transformed and quantized data, as well as the motion vector data form the video coding layer (VCL). The next data layer is known as the network adaptation layer (NAL) which adds the header information and formats the VCL representation of the video so as to convey the data through he transport layer.

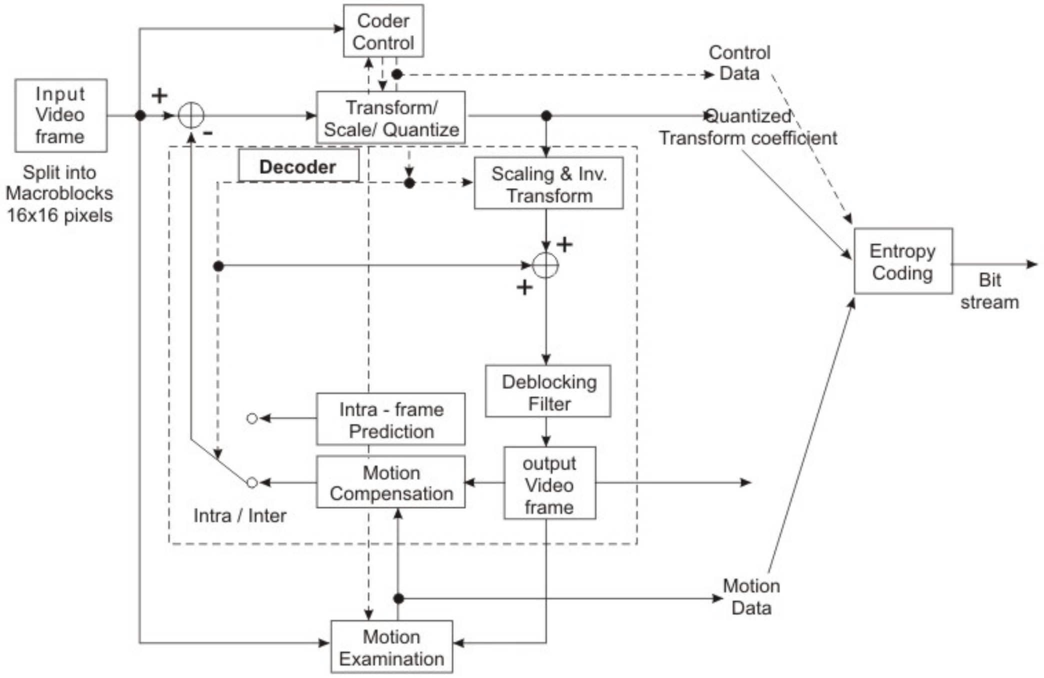


Fig. 27.6. Block diagram of H-264 codec.

The VCL follows a block-based hybrid video coding approach, as illustrated in the block diagram of H.264 codec in fig.27.6. The basic source coding algorithm combines exploitation of temporal redundancies through motion estimation and compensation and exploitation of spatial redundancies through integer transform.

Some of the major features of the VCL are listed below:

- A coded picture can either represent an entire frame or a field (*top field* of the *bottom field*)
- Uses Y-U-V colour representation with 4 : 2 : 0 sampling
- The picture us divided into fixed-size macroblocks of 16 x 16 pixels, which are the basic building blocks for the H.264 codecs.
- A group of macroblocks form slices and a picture may contain one or several slices
- Adaptive frame/field operation.

The first two features are also supported in MPEG-2 standard. *Slice* and *flexible macroblock ordering* (FMO) are the new concepts in H.264 and will be explained in the following subsection.

27.4.1 Slices and FMO

A slice is composed of a sequence of macroblocks in the raster scan order when it is not using the FMO. The concept of a *slice* is illustrated in fig. 27.7.

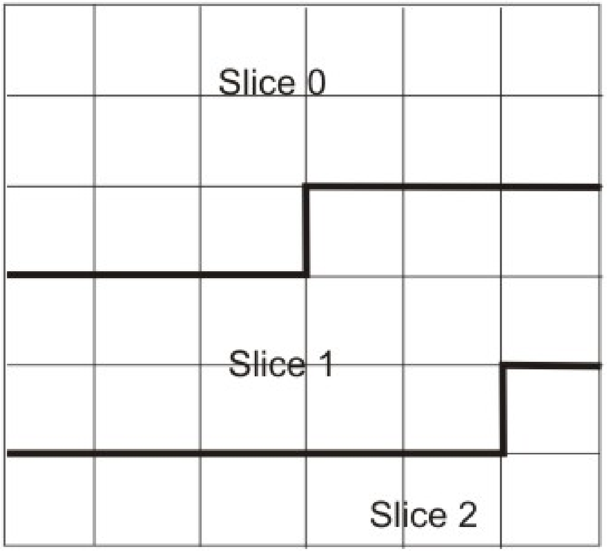


Fig. 27.7 Subdivision of a picture into slices , when not using the FMO.

In fig 27.7, we have shown that a picture is subdivided into macroblocks, shown as small squares. The illustration shows three slices (slice-0, slice –1 and slice-2).

Slices form independent entities and these can be encoded and decoded without any data reference from other slices. Each slice may be encoded into separate packets. The H.264 standard has introduced a *Flexible Macroblock Ordering* (FMO) concept, using which slice groups can be formed by mapping a macroblock into one of the slice group specified. Each slice group can be partitioned into one or more number of slices, such that slice is a sequence of macroblock in the same slice group. Fig 27.8 illustrates the slice group concept, using the FMO. In fig 27.8(a), each macroblock is mapped into one of the three slice group (# 0, # 1 and #2) by the FMO. Slice group # 0 and slice group # 1 belong to two foreground regions, whereas the slice group # 2 belongs to the leftover, or the background regions. The slice groups may not be composed of a sequence of macroblocks from the original picture. For example, in fig 27.8(b), the slice group have a checker board type mappings.

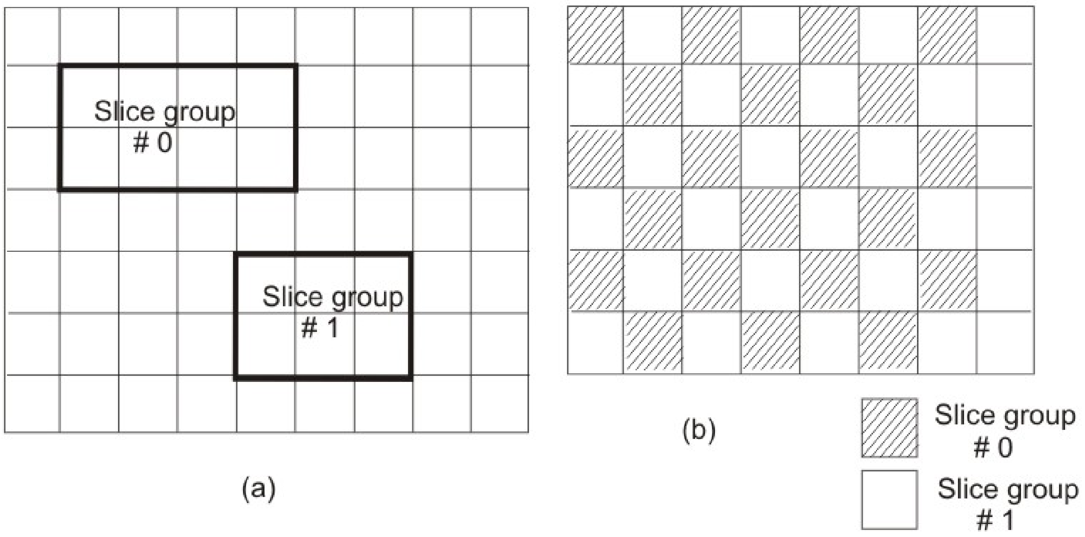


Fig. 27.8. Slice group formation usinf FMO.

Each slice may be coded using one of the following coding types :

- ***I-Slice***: where each macroblock within a slice are coded with intra prediction.
- ***P-Slice:*** In addition to the coding types of *I-slice*, some macroblocks can be coded using forward inter prediction.
- ***B-slice:*** In addition to the coding types of *P-slice*, some macroblocks can be coded through bi-directional prediction.
- ***SP-slice:*** Known as switching P-slice, where efficient switching between different pre-coded pictures is possible.
- ***SI-slice:*** Known as switching I-slice. This allows an exact match of a macroblock in an SP-slice for random access and error recovery.

27.5 Network Adaptations Layer (NAL)

The NAL provides a network friendly interface to the VCL to suit a broad variety of systems. It facilitates mapping of VCL data to transport layers such as

- RTP/IP for real time wire line and wireless internet protocols.
- File formats for storage
- H.32 X for wireline and wireless conversational service.

- MPEG-2 systems for broadcast service.

The coded video data is organized into NAL units, each of which is effectively a packet containing an integer number of bytes. The NAL unit structure is suitable for use in byte stream format, as well as packet oriented transport systems like RTP.

For a more detailed treatment on NAL, the reader is referred to Wenger [1].

27.6 Conclusion

This lesson provided a broad overview of the latest video coding standard H.264. More application areas are emerging and it is expected that over the next few years, H.264 will dominate both wireline and wireless video coding applications.