# Chapter 11
# MPEG Video Coding — MPEG-1, 2, 4 and 7

# 11.1 Overview

- **MPEG**: *Moving Pictures Experts Group*, established in 1988 for the development of digital video.
- It is appropriately recognized that proprietary interests need to be maintained within the family of MPEG standards:
  - Accomplished by defining only a compressed bitstream that implicitly defines the decoder.
  - The compression algorithms, and thus the encoders, are completely up to the manufacturers.

# 11.2 MPEG-1

- MPEG-1 adopts the CCIR601 digital TV format also known as SIF (*Source Input Format*).

- MPEG-1 supports only non-interlaced video. Normally, its picture resolution is:
  - 352 × 240 for NTSC video at 30 fps
  - 352 × 288 for PAL video at 25 fps
  - It uses 4:2:0 chroma subsampling

- The MPEG-1 standard is also referred to as ISO/IEC 11172. It has five parts: 11172-1 Systems, 11172-2 Video, 11172-3 Audio, 11172-4 Conformance, and 11172-5 Software.

# Motion Compensation in MPEG-1

- Motion Compensation (MC) based video encoding in H.261 works as follows:

  - In Motion Estimation (ME), each macroblock (MB) of the Target P-frame is assigned a best matching MB from the previously coded I or P frame - **prediction**.

  - **prediction error**: The difference between the MB and its matching MB, sent to DCT and its subsequent encoding steps.

  - The prediction is from a previous frame — **forward prediction**.

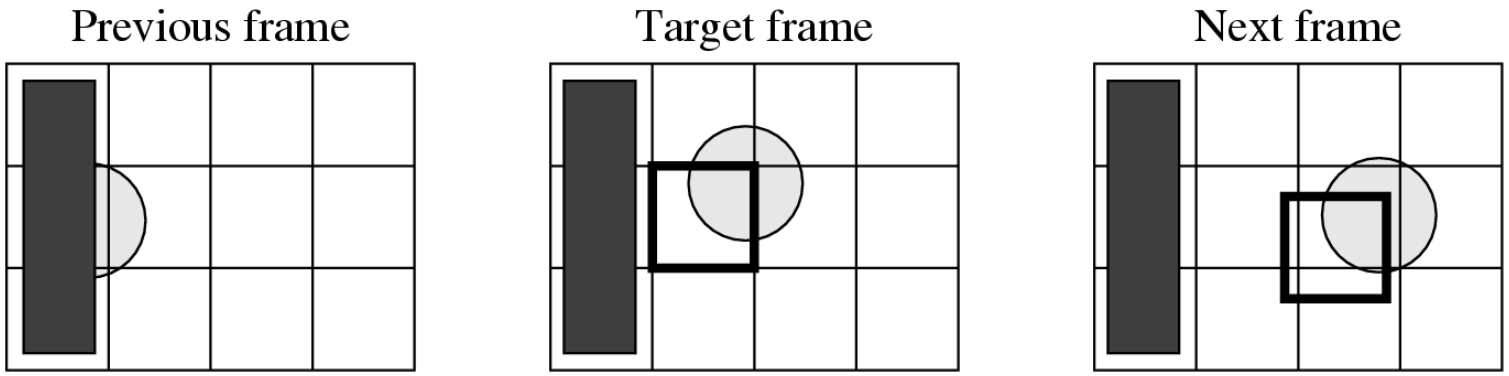



**Fig 11.1:** The Need for Bidirectional Search.

The MB containing part of a ball in the Target frame cannot find a good matching MB in the previous frame because half of the ball was occluded by another object. A match however can readily be obtained from the next frame.

# Motion Compensation in MPEG-1 (Cont'd)

- MPEG introduces a third frame type — *B-frames*, and its accompanying bi-directional motion compensation.
- The MC-based B-frame coding idea is illustrated in Fig. 11.2:
  - Each MB from a B-frame will have up to *two* motion vectors (MVs) (one from the forward and one from the backward prediction).
  - If matching in both directions is successful, then two MVs will be sent and the two corresponding matching MBs are averaged (indicated by '%' in the figure) before comparing to the Target MB for generating the prediction error.
  - If an acceptable match can be found in only one of the reference frames, then only one MV and its corresponding MB will be used from either the forward or backward prediction.

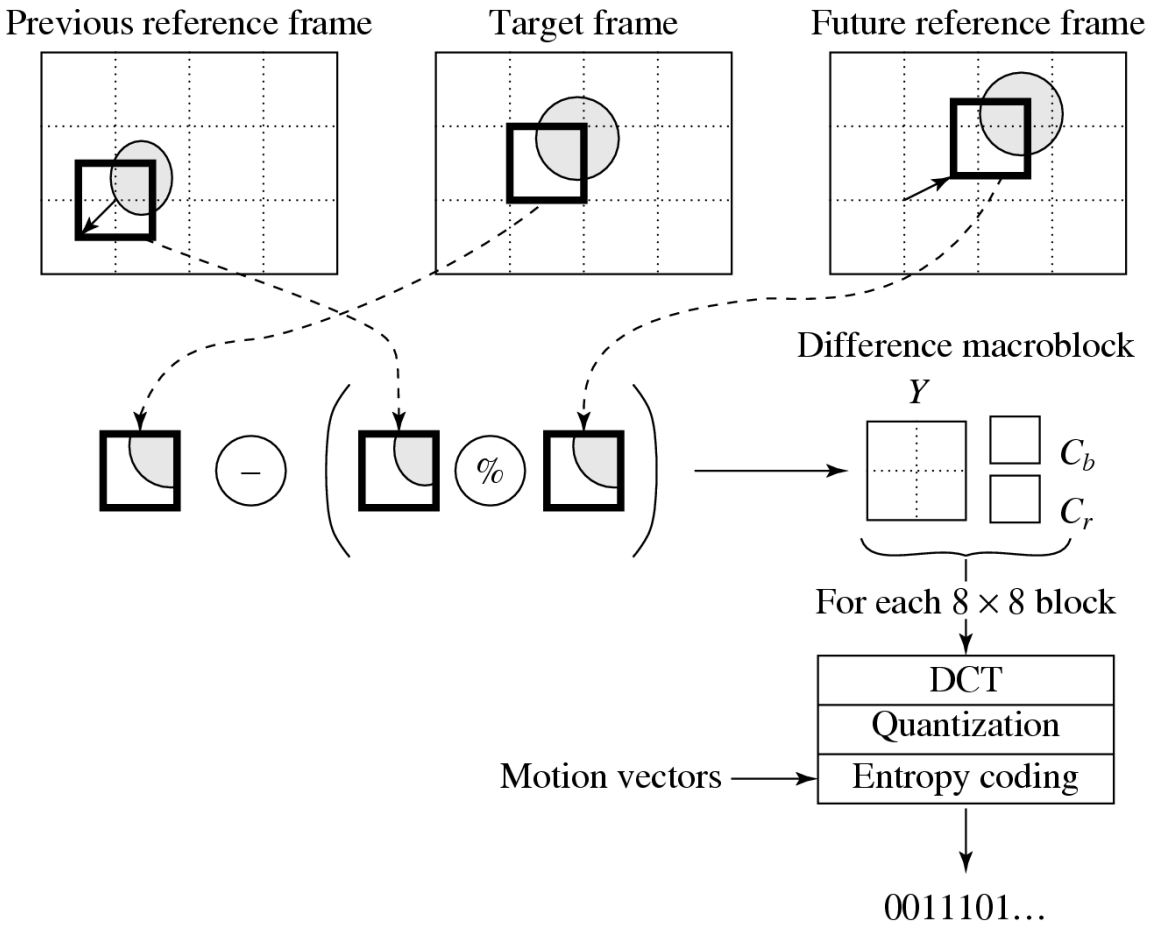



**Fig 11.2:** B-frame Coding Based on Bidirectional Motion Compensation.

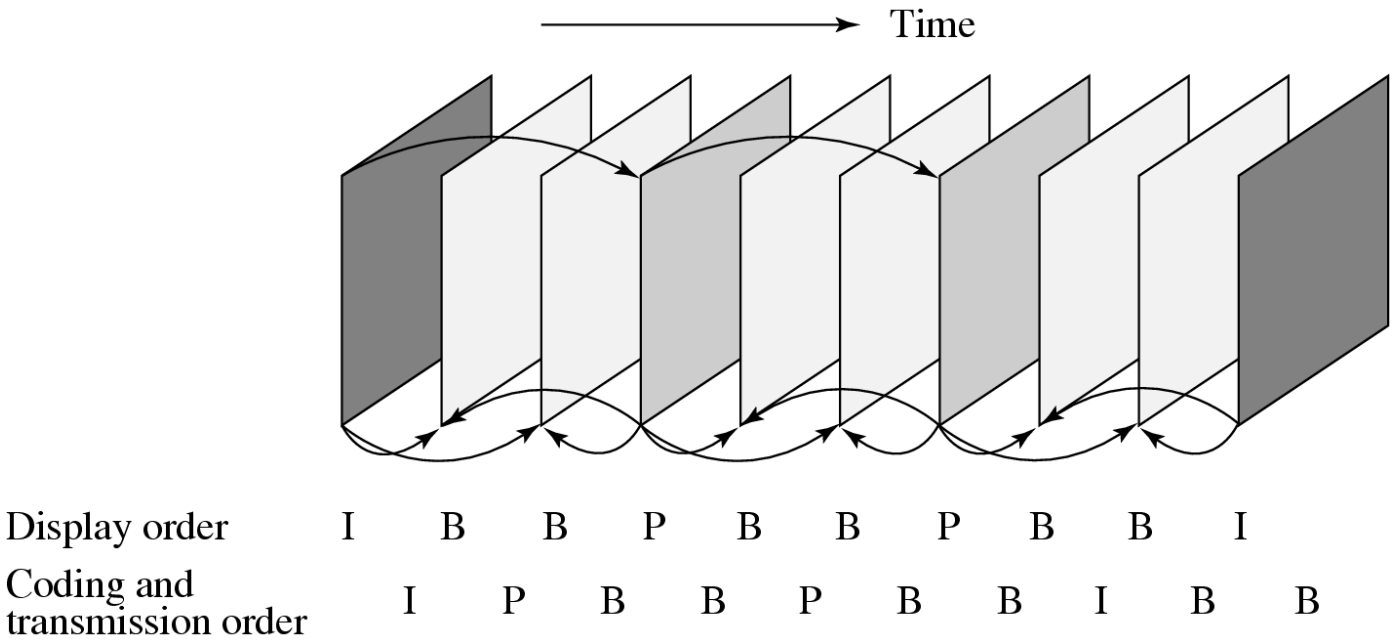



**Fig 11.3:** MPEG Frame Sequence.

# Other Major Differences from H.261

- Source formats supported:
  - H.261 only supports CIF (352 × 288) and QCIF (176 × 144) source formats, MPEG-1 supports SIF (352 × 240 for NTSC, 352 × 288 for PAL).
  - MPEG-1 also allows specification of other formats as long as the Constrained Parameter Set (CPS) as shown in Table 11.1 is satisfied:

| Parameter | Value |
|---|---|
| Horizontal size of picture | ≤ 768 |
| Vertical size of picture | ≤ 576 |
| No. of MBs / picture | ≤ 396 |
| No. of MBs / second | ≤ 9,900 |
| Frame rate | ≤ 30 fps |
| Bit-rate | ≤ 1,856 kbps |

# Other Major Differences from H.261 (Cont'd)

- Instead of GOBs as in H.261, an MPEG-1 picture can be divided into one or more **slices** (Fig. 11.4):
  - May contain variable numbers of macroblocks in a single picture.
  - May also start and end anywhere as long as they fill the whole picture.
  - Each slice is coded independently — additional flexibility in bit-rate control.
  - Slice concept is important for error recovery.

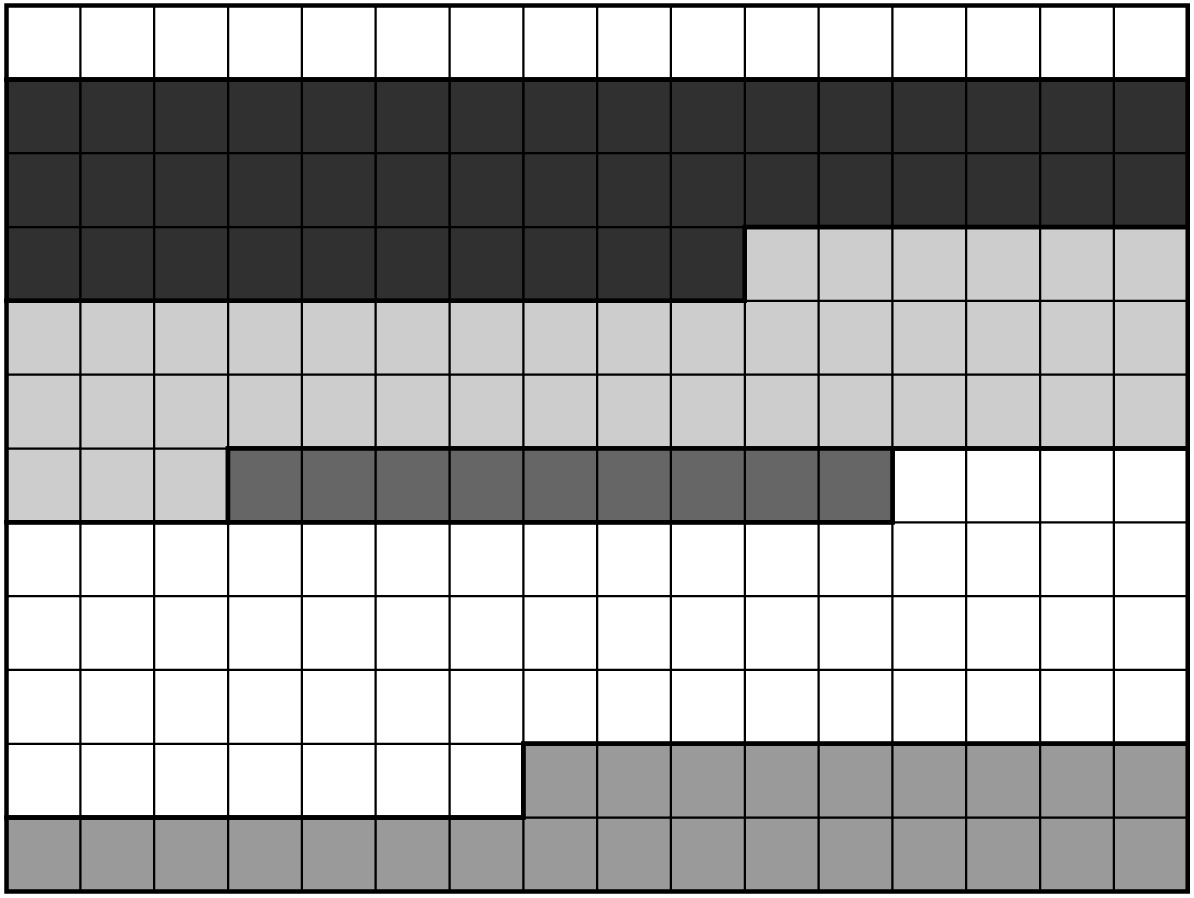


**Fig 11.4:** Slices in an MPEG-1 Picture.

# Other Major Differences from H.261 (Cont'd)

- Quantization:
  - MPEG-1 quantization uses different quantization tables for its Intra and Inter coding (Table 11.2 and 11.3).

For DCT coefficients in Intra mode:

$$QDCT[i,j] = round\left(\frac{8\times DCT[i,j]}{step\_size[i,j]}\right) = round\left(\frac{8\times DCT[i,j]}{Q_1[i,j]*scale}\right) \quad (11.1)$$

For DCT coefficients in Inter mode:

$$QDCT[i,j] = \left\lfloor \frac{8\times DCT[i,j]}{step\_size[i,j]} \right\rfloor = \left\lfloor \frac{8\times DCT[i,j]}{Q_2[i,j]*scale} \right\rfloor \quad (11.2)$$

**Table 11.2:** Default Quantization Table ($Q_1$) for Intra-Coding

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 8 | 16 | 19 | 22 | 26 | 27 | 29 | 34 |
| 16 | 16 | 22 | 24 | 27 | 29 | 34 | 37 |
| 19 | 22 | 26 | 27 | 29 | 34 | 34 | 38 |
| 22 | 22 | 26 | 27 | 29 | 34 | 37 | 40 |
| 22 | 26 | 27 | 29 | 32 | 25 | 40 | 48 |
| 26 | 27 | 29 | 32 | 35 | 40 | 48 | 58 |
| 26 | 27 | 29 | 34 | 38 | 46 | 56 | 69 |
| 27 | 29 | 35 | 38 | 46 | 56 | 69 | 83 |

**Table 11.3:** Default Quantization Table ($Q_2$) for Inter-Coding

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 16 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |

# Other Major Differences from H.261 (Cont'd)

- MPEG-1 allows motion vectors to be of sub-pixel precision (1/2 pixel). The technique of "bilinear interpolation" for H.263 can be used to generate the needed values at half-pixel locations.

- Compared to the maximum range of ±15 pixels for motion vectors in H.261, MPEG-1 supports a range of [−512, 511.5] for half-pixel precision and [−1,024, 1,023] for full-pixel precision motion vectors.

- The MPEG-1 bitstream allows random access — accomplished by GOP layer in which each GOP is time coded.

# Typical Sizes of MPEG-1 Frames

- The typical size of compressed P-frames is significantly smaller than that of I-frames — because temporal redundancy is exploited in inter-frame compression.

- B-frames are even smaller than P-frames — because of (a) the advantage of bi-directional prediction and (b) the lowest priority given to B-frames.

**Table 11.4**: Typical Compression Performance of MPEG-1 Frames

| Type | Size | Compression |
|---|---|---|
| I | 18kB | 7:1 |
| P | 6kB | 20:1 |
| B | 2.5kB | 50:1 |
| Avg | 4.8kB | 27:1 |

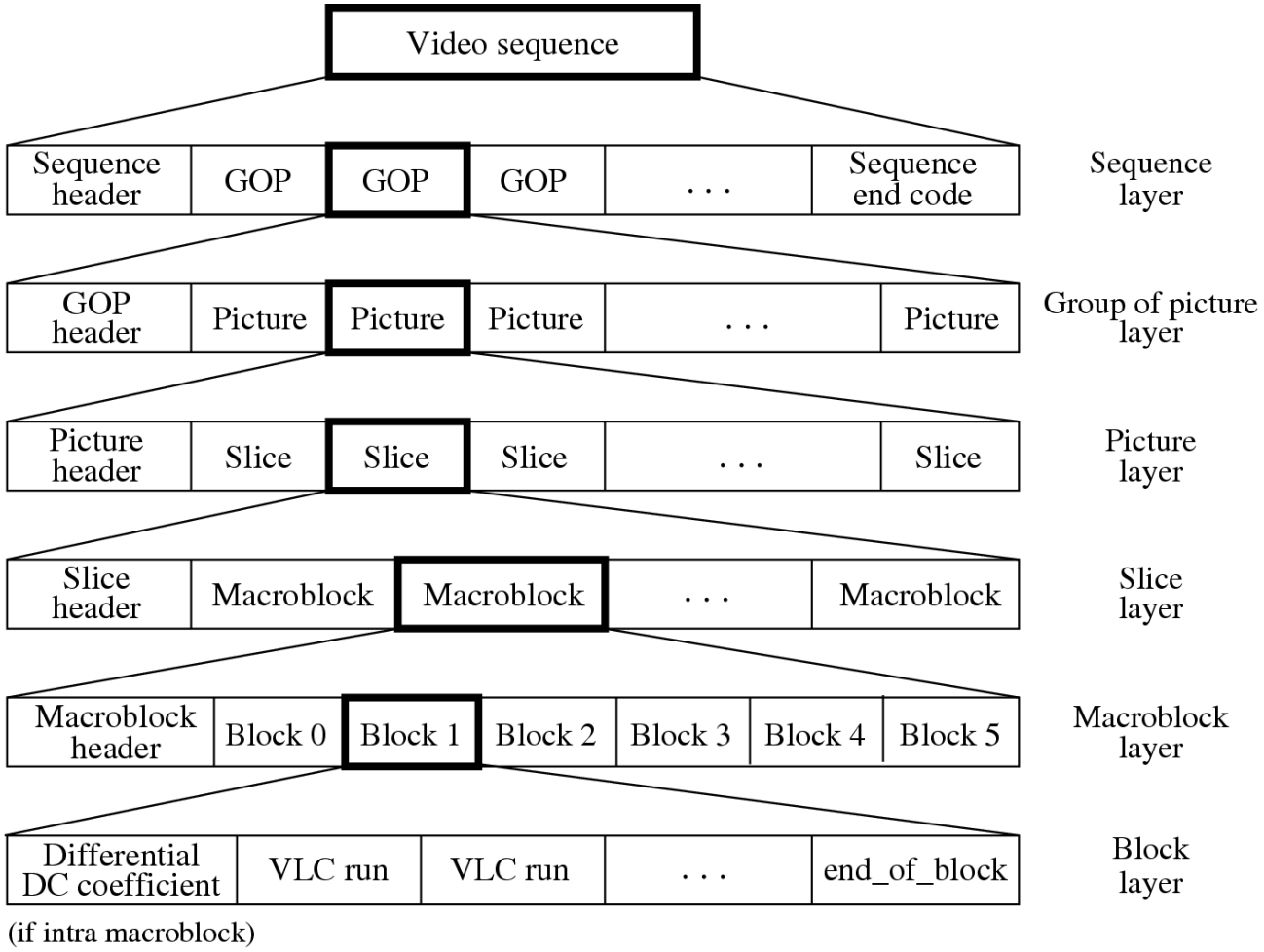



**Fig 11.5:** Layers of MPEG-1 Video Bitstream.

# 11.3 MPEG-2

- **MPEG-2**: For higher quality video at a bit-rate of more than 4 Mbps.
- Defined seven **profiles** aimed at different applications
  - **Simple, Main, SNR scalable, Spatially scalable, High, 4:2:2, Multiview**.
  - Within each profile, up to four *levels* are defined (Table 11.5).
  - The DVD video specification allows only four display resolutions: 720×480, 704×480, 352×480, and 352×240
  - a restricted form of the MPEG-2 Main profile at the Main and Low levels.

**Table 11.5:** Profiles and Levels in MPEG-2

| Level | Simple profile | Main profile | SNR Scalable profile | Spatially Scalable profile | High Profile | 4:2:2 Profile | Multiview Profile |
|---|---|---|---|---|---|---|---|
| High | | * | | | * | | |
| High | | * | | * | * | | |
| 1440 | * | * | * | | * | * | * |
| Main | | * | * | | | | |
| Low | | | | | | | |

**Table 11.6:** Four Levels in the Main Profile of MPEG-2

| Level | Max. Resolution | Max fps | Max pixels/sec | Max coded Data Rate (Mbps) | Application |
|---|---|---|---|---|---|
| High | 1920 × 1152 | 60 | $62.7 \times 10^6$ | 80 | film production |
| High 1440 | 1440 × 1152 | 60 | $47.0 \times 10^6$ | 60 | consumer HDTV |
| Main | 720 × 576 | 30 | $10.4 \times 10^6$ | 15 | Studio TV |
| Low | 352 × 288 | 30 | $3.0 \times 10^6$ | 4 | consumer tape equiv. |

# Supporting Interlaced Video

- MPEG-2 must support interlaced video as well since this is one of the options for digital broadcast TV and HDTV.

- In interlaced video each frame consists of two fields, referred to as the *top-field* and the *bottom-field*.

  - In a *Frame-picture*, all scanlines from both fields are interleaved to form a single frame, then divided into 16×16 macroblocks and coded using MC.

  - If each field is treated as a separate picture, then it is called *Field-picture*.

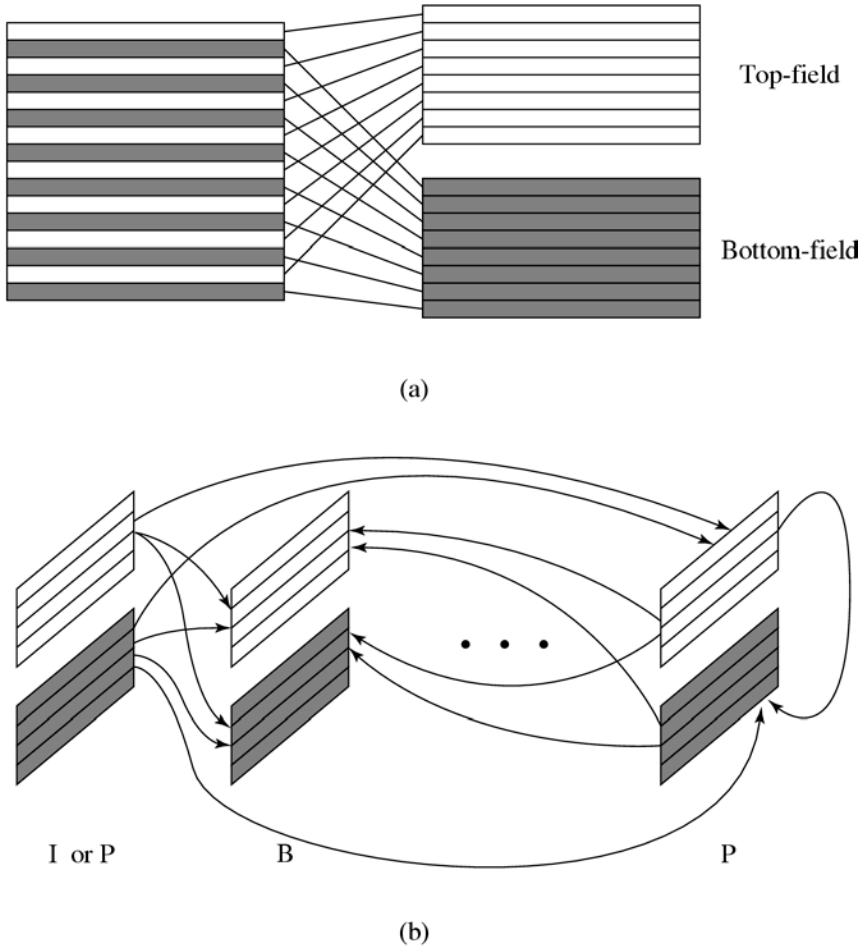



**Fig. 11.6:** Field pictures and Field-prediction for Field-pictures in MPEG-2.

(a) Frame–picture vs. Field–pictures

(b) Field Prediction for Field–pictures

# Five Modes of Predictions

- MPEG-2 defines **Frame Prediction** and **Field Prediction** as well as five prediction modes:

  1. **Frame Prediction for Frame-pictures**: Identical to MPEG-1 MC-based prediction methods in both P-frames and B-frames.
  2. **Field Prediction for Field-pictures**: A macroblock size of 16 × 16 from Field-pictures is used. For details, see Fig. 11.6(b).

3. **Field Prediction for Frame-pictures**: The top-field and bottom-field of a Frame-picture are treated separately. Each 16 × 16 macroblock (MB) from the target Frame-picture is split into two 16 × 8 parts, each coming from one field. Field prediction is carried out for these 16 × 8 parts in a manner similar to that shown in Fig. 11.6(b).

4. 16×8 **MC for Field-pictures**: Each 16×16 macroblock (MB) from the target Field-picture is split into top and bottom 16 × 8 halves. Field prediction is performed on each half. This generates two motion vectors for each 16×16 MB in the P-Field-picture, and up to four motion vectors for each MB in the B-Field-picture.

   This mode is good for a finer MC when motion is rapid and irregular.

5. **Dual-Prime for P-pictures**: First, Field prediction from each previous field with the same parity (top or bottom) is made. Each motion vector **mv** is then used to derive a calculated motion vector **cv** in the field with the opposite parity taking into account the temporal scaling and vertical shift between lines in the top and bottom fields. For each MB the pair **mv** and **cv** yields two preliminary predictions. Their prediction errors are averaged and used as the final prediction error.

This mode mimics B-picture prediction for P-pictures without adopting backward prediction (and hence with less encoding delay).

This is the only mode that can be used for either Frame-pictures or Field-pictures.

# Alternate Scan and Field DCT

- Techniques aimed at improving the effectiveness of DCT on prediction errors, only applicable to Frame-pictures in interlaced videos:
  - Due to the nature of interlaced video the consecutive rows in the 8×8 blocks are from different fields, there exists less correlation between them than between the alternate rows.
  - Alternate scan recognizes the fact that in interlaced video the vertically higher spatial frequency components may have larger magnitudes and thus allows them to be scanned earlier in the sequence.
- In MPEG-2, **Field_DCT** can also be used to address the same issue.

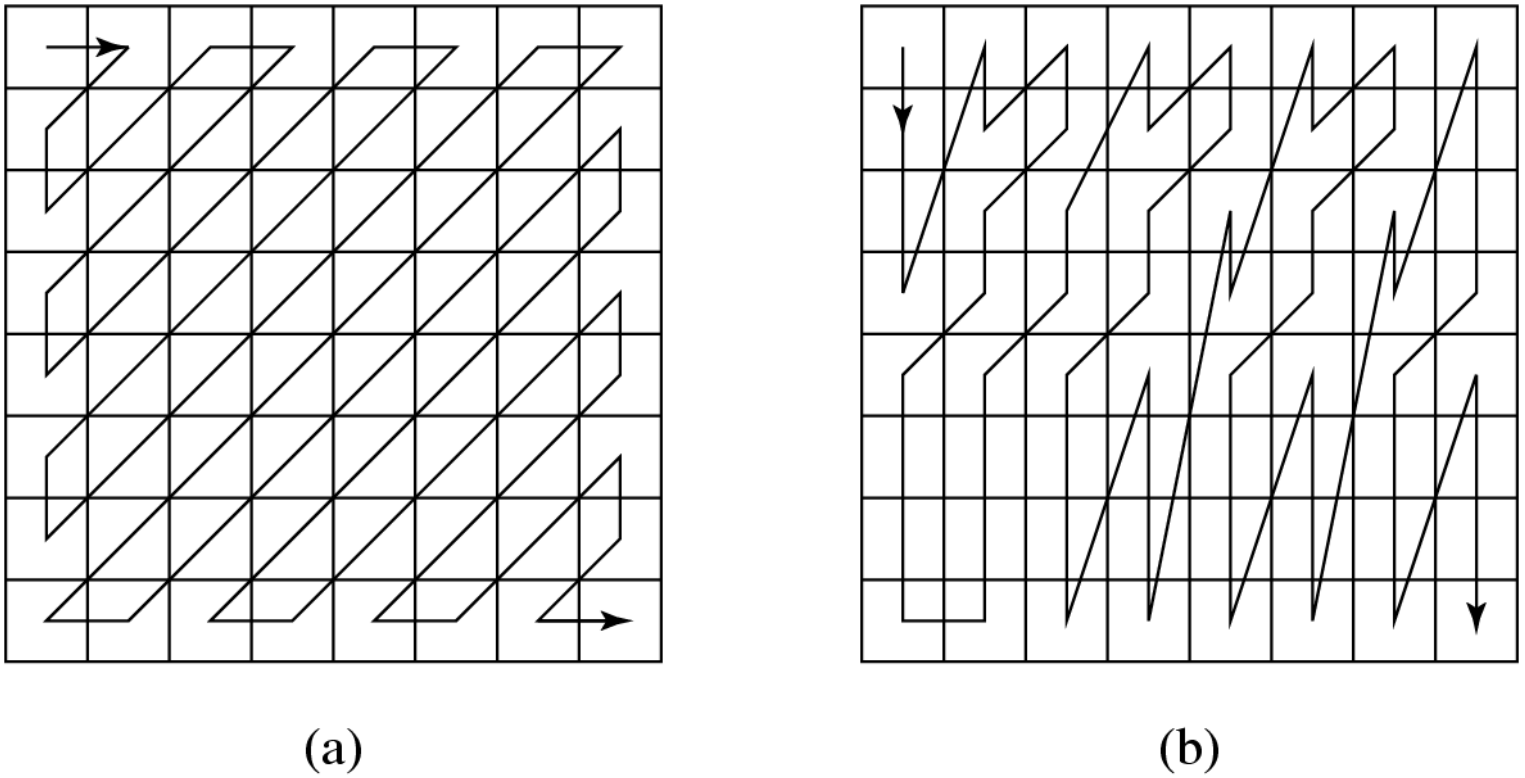



**Fig 11.7:** Zigzag and Alternate Scans of DCT Coefficients for Progressive and Interlaced Videos in MPEG-2.

# MPEG-2 Scalabilities

- The MPEG-2 **scalable coding**: A base layer and one or more enhancement layers can be defined — also known **as layered coding**.
  - The base layer can be independently encoded, transmitted and decoded to obtain basic video quality.
  - The encoding and decoding of the enhancement layer is dependent on the base layer or the previous enhancement layer.

- Scalable coding is especially useful for MPEG-2 video transmitted over networks with following characteristics:
  - Networks with very different bit-rates.
  - Networks with variable bit rate (VBR) channels.
  - Networks with noisy connections.

# MPEG-2 Scalabilities (Cont'd)

- MPEG-2 supports the following scalabilities:

1. SNR Scalability—enhancement layer provides higher SNR.
2. Spatial Scalability — enhancement layer provides higher spatial resolution.
3. Temporal Scalability—enhancement layer facilitates higher frame rate.
4. Hybrid Scalability — combination of any two of the above three scalabilities.
5. Data Partitioning — quantized DCT coefficients are split into partitions.

# SNR Scalability

- **SNR scalability**: Refers to the enhencement/refinement over the base layer to improve the Signal-Noise-Ratio (SNR).

- The MPEG-2 SNR scalable encoder will generate output bitstreams *Bits_base* and *Bits_enhance* at two layers:

  1. At the Base Layer, a coarse quantization of the DCT coefficients is employed which results in fewer bits and a relatively low quality video.
  2. The coarsely quantized DCT coefficients are then inversely quantized ($Q^{-1}$) and fed to the Enhancement Layer to be compared with the original DCT coefficient.
  3. Their difference is finely quantized to generate a **DCT coefficient refinement**, which, after VLC, becomes the bitstream called Bits_enhance.

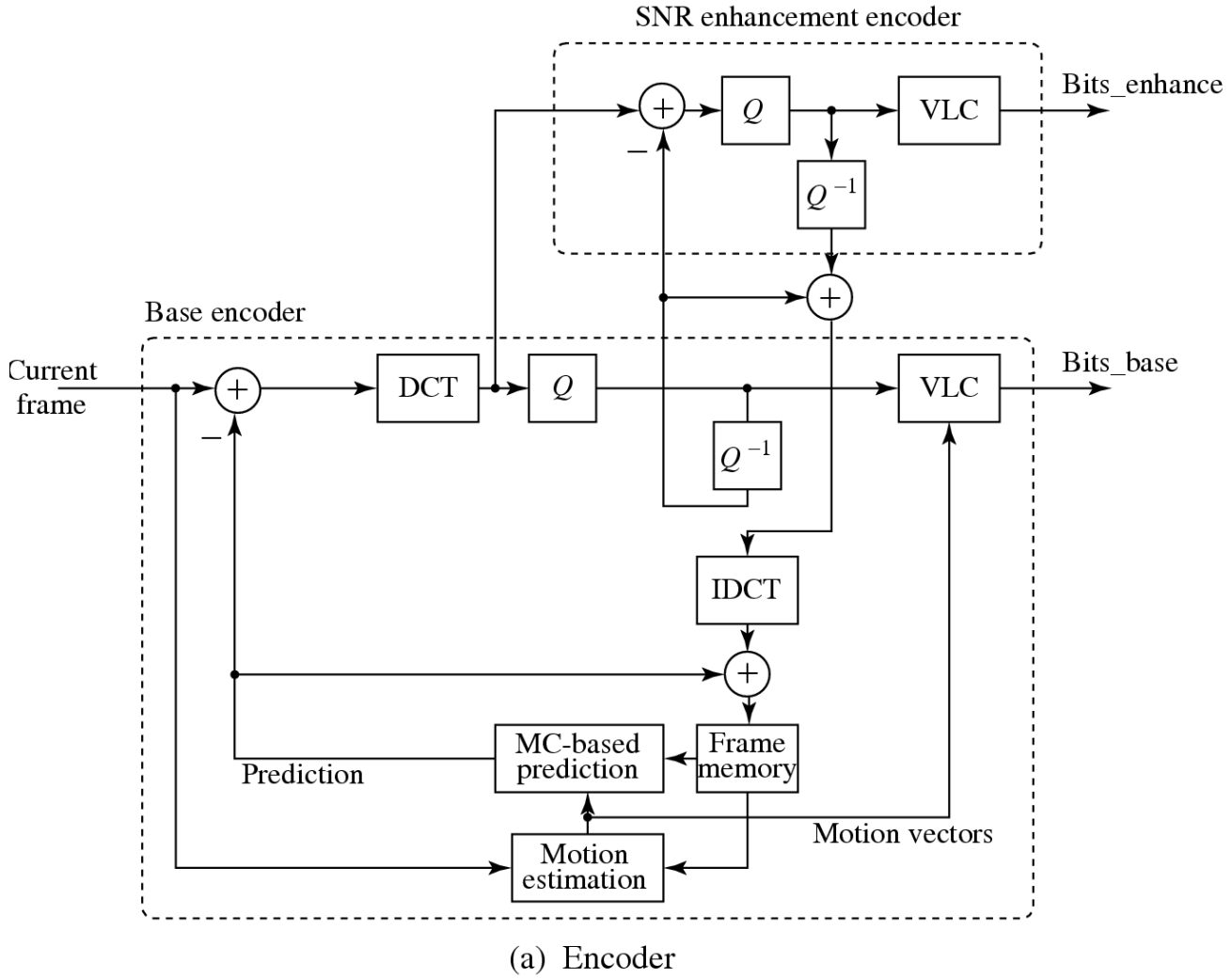



**Fig 11.8 (a):** MPEG-2 SNR Scalability (Encoder).

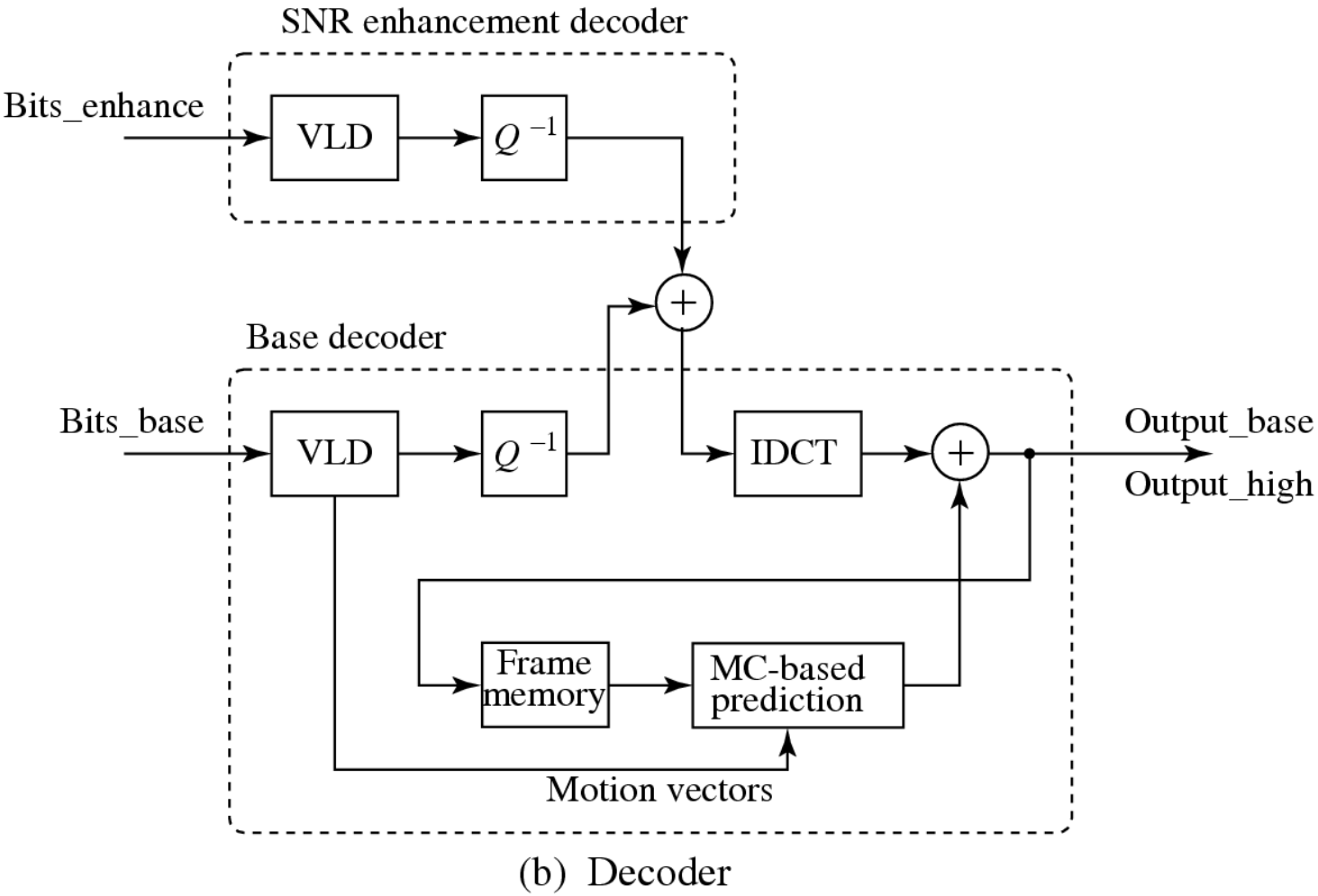



**Fig 11.8 (b):** MPEG-2 SNR Scalability (Decoder).

# Spatial Scalability

- The base layer is designed to generate bitstream of reduced resolution pictures. When combined with the enhancement layer, pictures at the original resolution are produced.
- The Base and Enhancement layers for MPEG-2 spatial scalability are not as tightly coupled as in SNR scalability.
- Fig. 11.9(a) shows a typical block diagram. Fig. 11.9(b) shows a case where temporal and spatial predictions are combined.

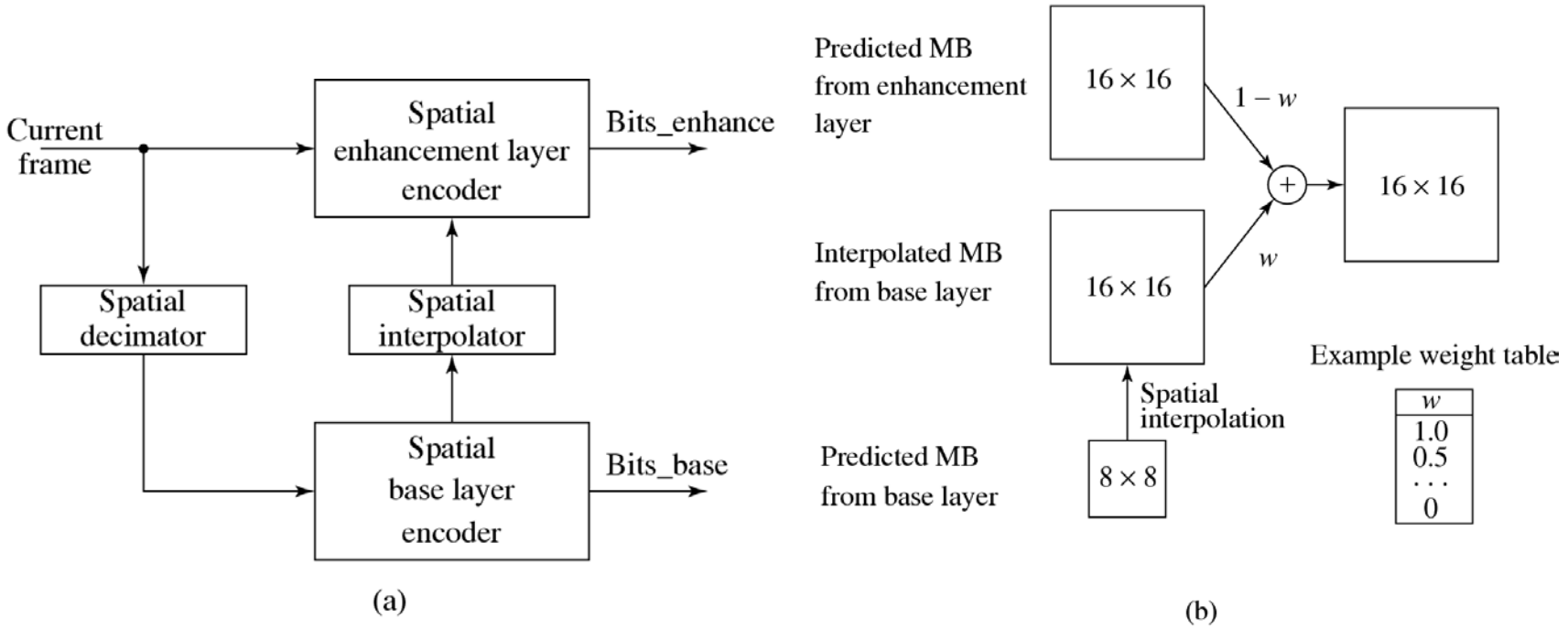
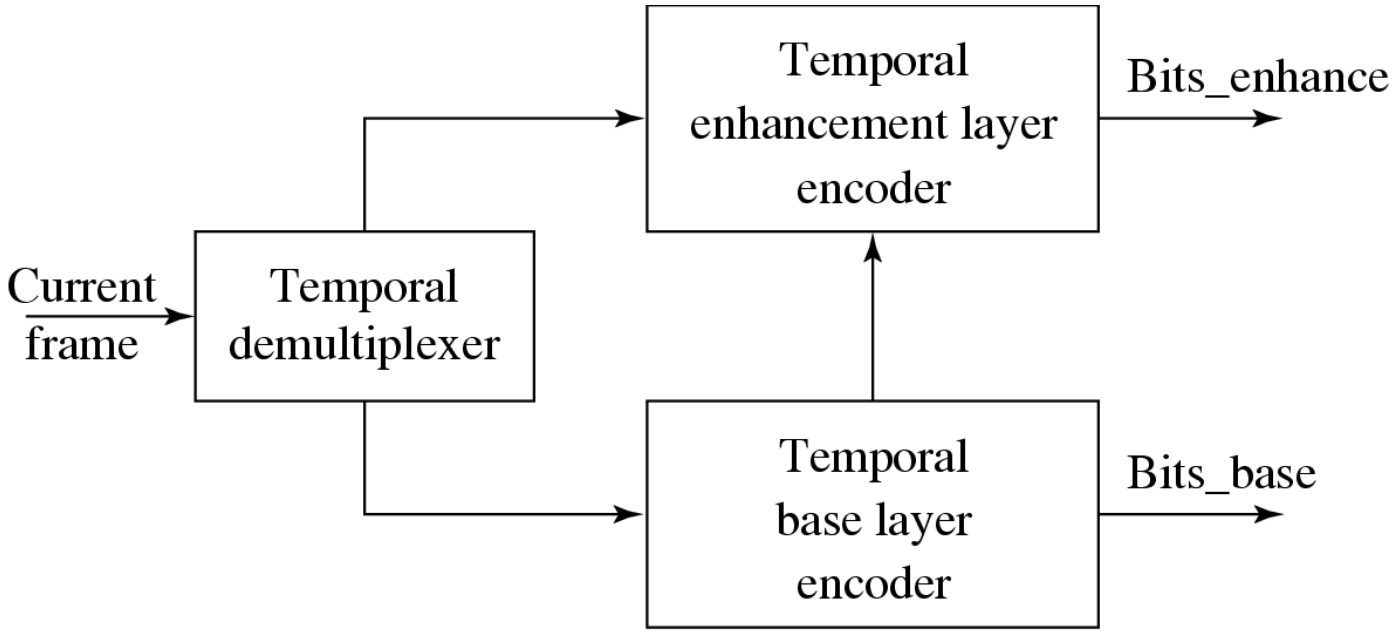



**Fig. 11.9:** Encoder for MPEG-2 Spatial Scalability.

(a) Block Diagram. (b) Combining Temporal and Spatial Predictions for Encoding at Enhancement Layer.

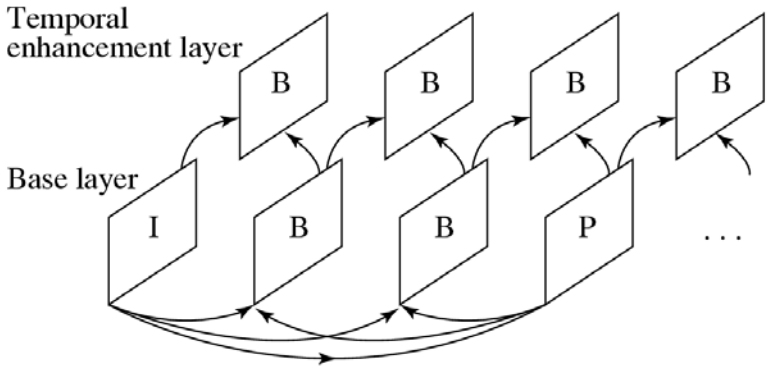
# Temporal Scalability

- The input video is temporally demultiplexed into two pieces, each carrying half of the original frame rate.
- Base Layer Encoder carries out the normal single-layer coding procedures for its own input video and yields the output bitstream Bits_base.
- The prediction of matching MBs at the Enhancement Layer can be obtained in two ways:
  - Interlayer MC (Motion-Compensated) Prediction (Fig. 11.10(b))
  - Combined MC Prediction and Interlayer MC Prediction (Fig. 11.10(c))

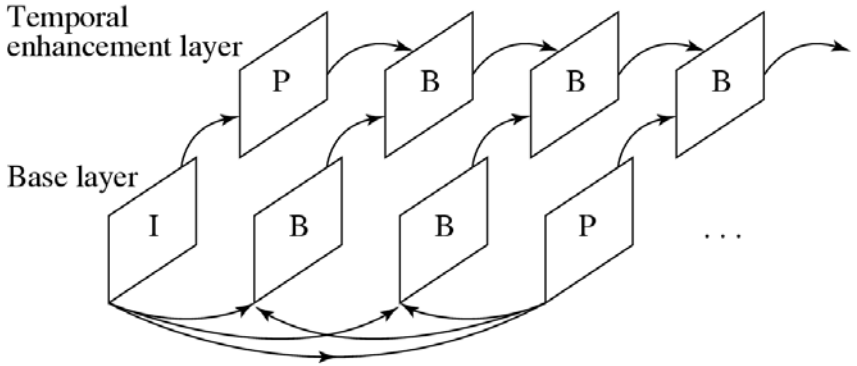



(a) Block Diagram

**Fig 11.10:** Encoder for MPEG-2 Temporal Scalability.




(b) Interlayer Motion-Compensated (MC) Prediction




(c) Combined MC Prediction and Interlayer MC Prediction

**Fig 11.10 (Cont'd):** Encoder for MPEG-2 Temporal Scalability

# Hybrid Scalability

- Any two of the above three scalabilities can be combined to form hybrid scalability:

  1. Spatial and Temporal Hybrid Scalability.
  2. SNR and Spatial Hybrid Scalability.
  3. SNR and Temporal Hybrid Scalability.

- Usually, a three-layer hybrid coder will be adopted which consists of Base Layer, Enhancement Layer 1, and Enhancement Layer 2.

# Data Partitioning

- The *Base partition* contains lower-frequency DCT coefficients, enhancement partition contains high-frequency DCT coefficients.
- Strictly speaking, data partitioning is not layered coding, since a single stream of video data is simply divided up and there is no further dependence on the base partition in generating the enhancement partition.
- Useful for transmission over noisy channels and for progressive transmission.

# Other Major Differences from MPEG-1

- **Better resilience to bit-errors**: In addition to *Program Stream*, a *Transport Stream* is added to MPEG-2 bit streams.
- **Support of 4:2:2 and 4:4:4 chroma subsampling**.
- **More restricted slice structure**: MPEG-2 slices must start and end in the same macroblock row. In other words, the left edge of a picture always starts a new slice and the longest slice in MPEG-2 can have only one row of macroblocks.
- **More flexible video formats**: It supports various picture resolutions as defined by DVD, ATV and HDTV.

# Other Major Differences from MPEG-1 (Cont'd)

- **Nonlinear quantization** — two types of scales are allowed:

  1. For the first type, scale is the same as in MPEG-1 in which it is an integer in the range of [1, 31] and $scale_i = i$.

  2. For the second type, a nonlinear relationship exists, i.e., $scale_i \neq i$. The $i$th scale value can be looked up from Table 11.7.

**Table 11.7:** Possible Nonlinear Scale in MPEG-2

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $scale_i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 |
| $i$ | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | |
| $scale_i$ | 28 | 32 | 36 | 40 | 44 | 48 | 52 | 56 | 64 | 72 | 80 | 88 | 96 | 104 | 112 | |

# 11.4.1 Overview of MPEG-4

- **MPEG-4**: a newer standard. Besides compression, pays great attention to issues about user interactivities.
- MPEG-4 departs from its predecessors in adopting a new **object-based coding**:
  - Offering higher compression ratio, also beneficial for digital video composition, manipulation, indexing, and retrieval.
  - Figure 11.11 illustrates how MPEG-4 videos can be composed and manipulated by simple operations on the visual objects.
- The bit-rate for MPEG-4 video now covers a large range between 5 kbps to 10 Mbps.

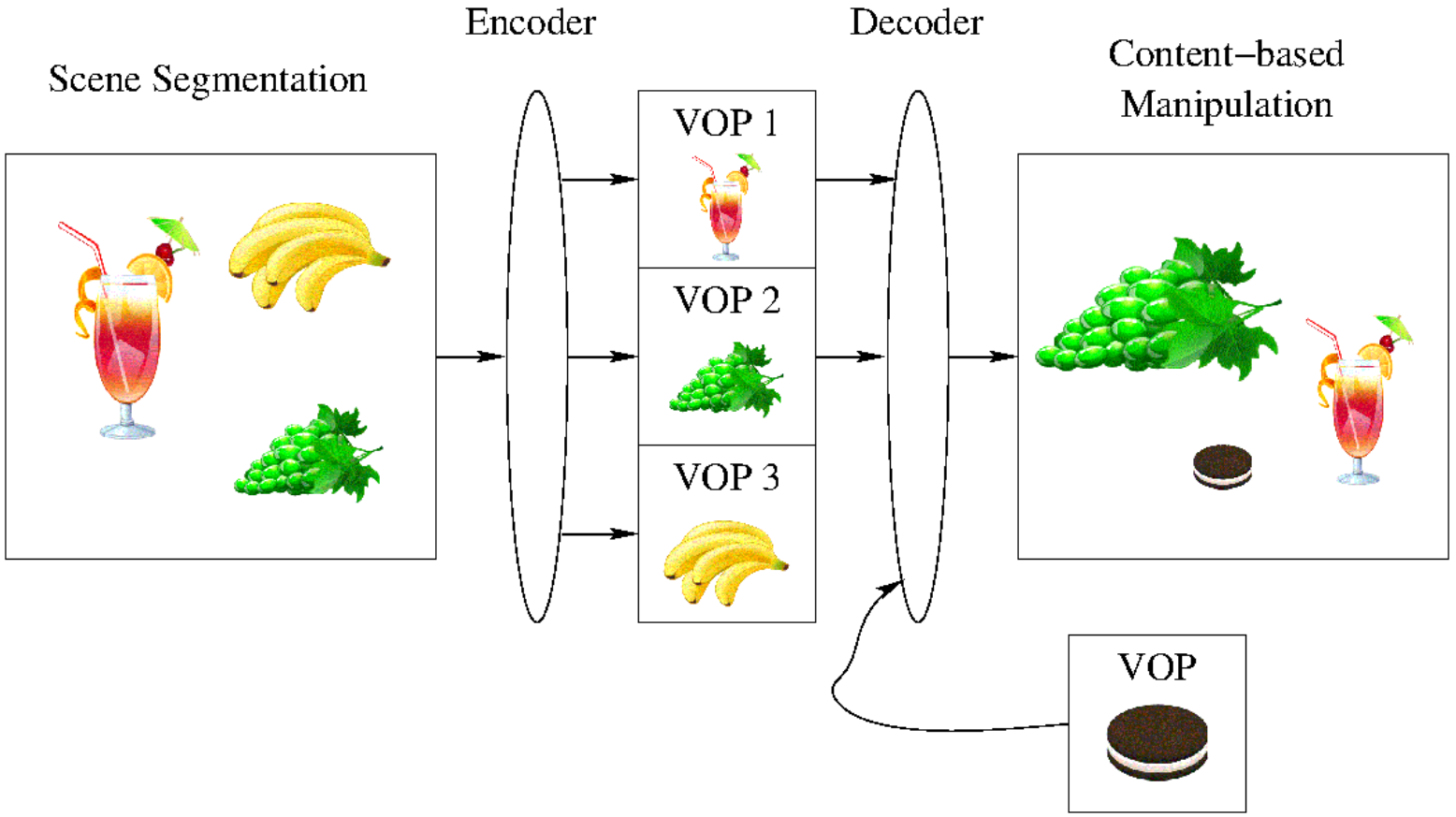



**Fig. 11.11:** Composition and Manipulation of MPEG-4 Videos. (VOP = Video object plane)

# Overview of MPEG-4 (Cont'd)

- MPEG-4 (Fig. 11.12(b)) is an entirely new standard for:

  (a) Composing media objects to create desirable audiovisual scenes.

  (b) Multiplexing and synchronizing the bitstreams for these media data entities so that they can be transmitted with guaranteed Quality of Service (QoS).

  (c) Interacting with the audiovisual scene at the receiving end — provides a toolbox of advanced coding modules and algorithms for audio and video compressions.

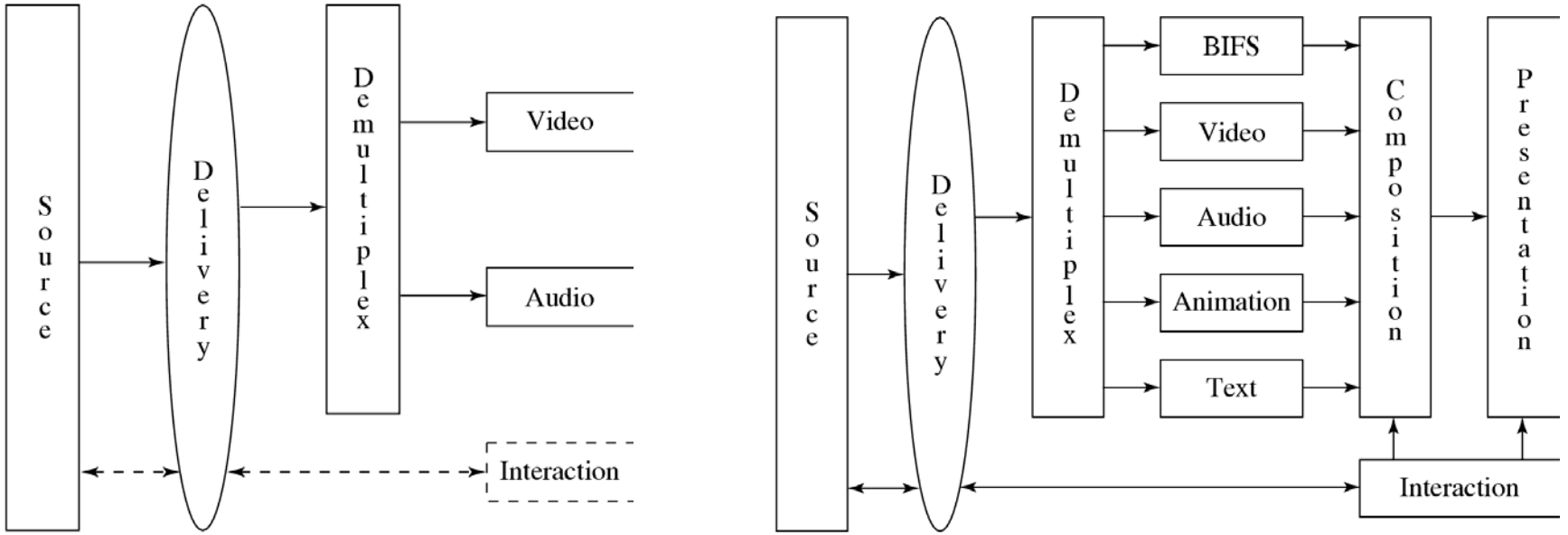



**Fig. 11.12:** Comparison of interactivities in MPEG standards:

(a) reference models in MPEG-1 and 2 (interaction in dashed lines supported only by MPEG-2);

(b) MPEG-4 reference model.

# Overview of MPEG-4 (Cont'd)

- The hierarchical structure of MPEG-4 visual bitstreams is very different from that of MPEG-1 and -2, it is very much video object-oriented.

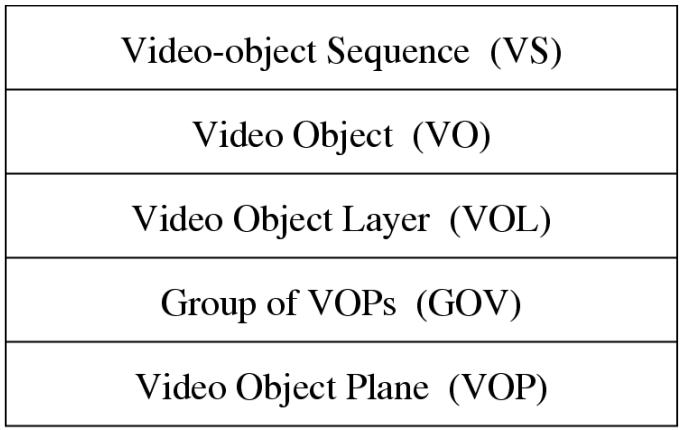



**Fig. 11.13:** Video Object Oriented Hierarchical Description of a Scene in MPEG-4 Visual Bitstreams.

# Overview of MPEG-4 (Cont'd)

1. **Video-object Sequence** (VS)—delivers the complete MPEG-4 visual scene, which may contain 2-D or 3-D natural or synthetic objects.

2. **Video Object** (VO) — a particular object in the scene, which can be of arbitrary (non-rectangular) shape corresponding to an object or background of the scene.

3. **Video Object Layer** (VOL) — facilitates a way to support (multi-layered) scalable coding. A VO can have multiple VOLs under scalable coding, or have a single VOL under non-scalable coding.

4. **Group of Video Object Planes** (GOV) — groups Video Object Planes together (optional level).

5. **Video Object Plane** (VOP) — a snapshot of a VO at a particular moment.

# 11.4.2 Video Object-based Coding in MPEG-4

## VOP-based vs. Frame-based Coding

- MPEG-1 and -2 do not support the VOP concept, and hence their coding method is referred to as **frame-based** (also known as **Block-based coding**).
- Fig. 11.14 (c) illustrates a possible example in which both potential matches yield small prediction errors for block-based coding.
- Fig. 11.14 (d) shows that each VOP is of arbitrary shape and ideally will obtain a unique motion vector consistent with the actual object motion.

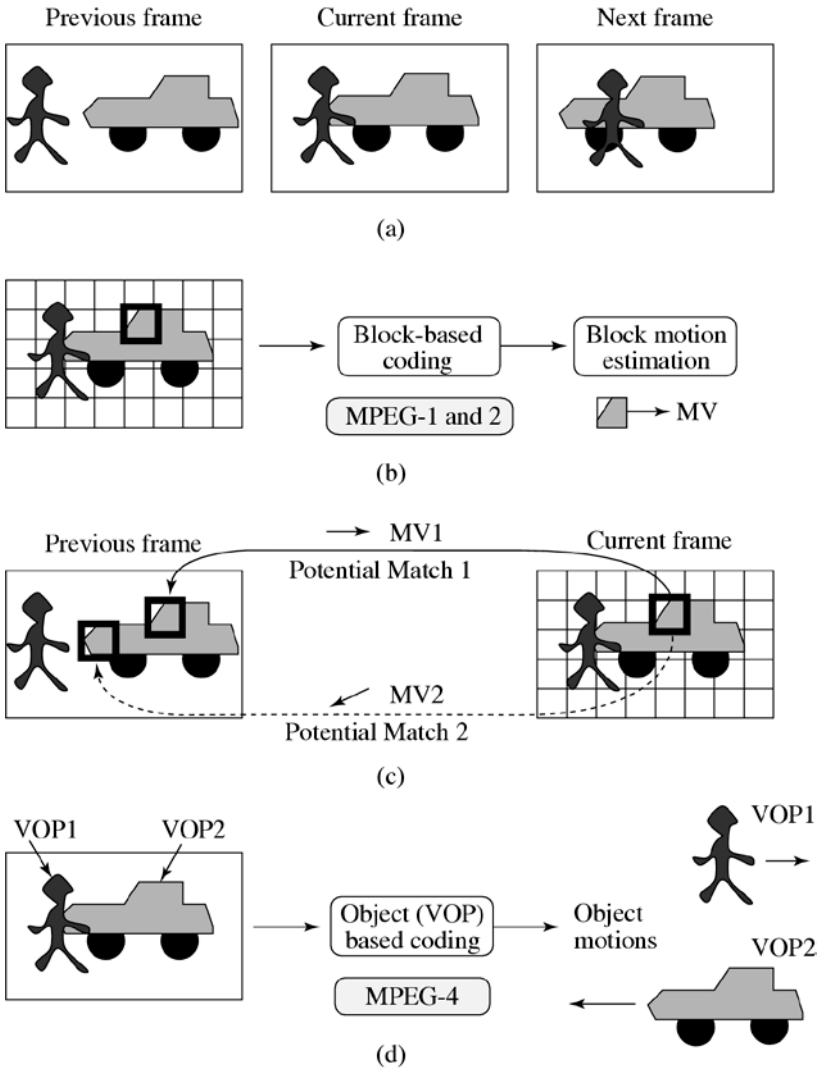



**Fig. 11.14:** Comparison between Block-based Coding and Object-based Coding.

# VOP-based Coding

- MPEG-4 VOP-based coding also employs the Motion Compensation technique:
  - An Intra-frame coded VOP is called an **I-VOP**.
  - The Inter-frame coded VOPs are called *P-VOPs* if only forward prediction is employed, or *B-VOPs* if bi-directional predictions are employed.
- The new difficulty for VOPs: may have arbitrary shapes, shape information must be coded in addition to the texture of the VOP.

Note: *texture* here actually refers to the visual content, that is the gray-level (or chroma) values of the pixels in the VOP.

# VOP-based Motion Compensation (MC)

- MC-based VOP coding in MPEG-4 again involves three steps:
  (a) Motion Estimation.
  (b) MC-based Prediction.
  (c) Coding of the prediction error.
- Only pixels within the VOP of the current (Target) VOP are considered for matching in MC.
- To facilitate MC, each VOP is divided into many macroblocks (MBs). MBs are by default 16×16 in luminance images and 8 × 8 in chrominance images.

- MPEG-4 defines a rectangular *bounding box* for each VOP (see Fig. 11.15 for details).

- The macroblocks that are entirely within the VOP are referred to as **Interior Macroblocks**.

- The macroblocks that straddle the boundary of the VOP are called **Boundary Macroblocks**.

To help matching every pixel in the target VOP and meet the mandatory requirement of rectangular blocks in transform codine (e.g., DCT), a pre-processing step of *padding* is applied to the Reference VOPs prior to motion estimation.

**Note**: Padding only takes place in the Reference VOPs.

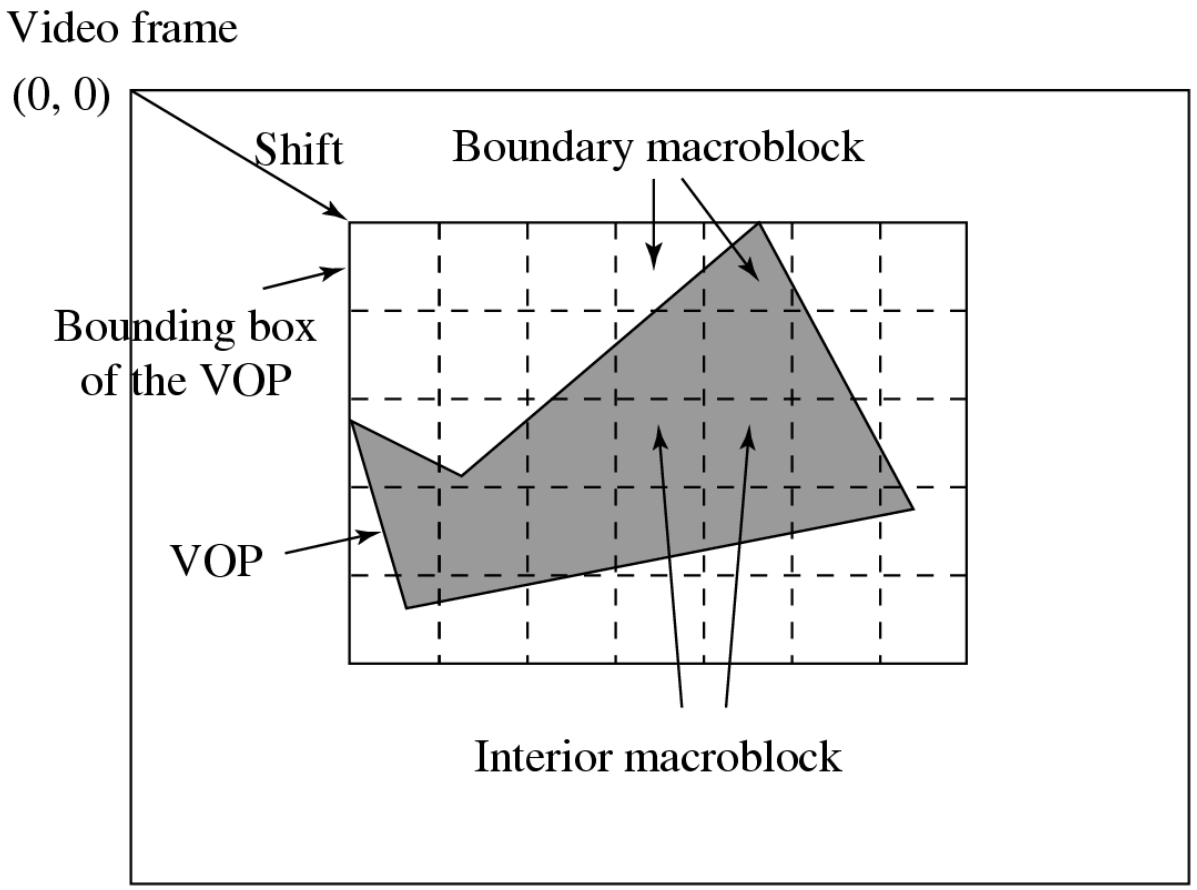



**Fig. 11.15:** Bounding Box and Boundary Macroblocks of VOP.

# Padding

- For all Boundary MBs in the Reference VOP, *Horizontal Repetitive Padding* is invoked first, followed by *Vertical Repetitive Padding*.

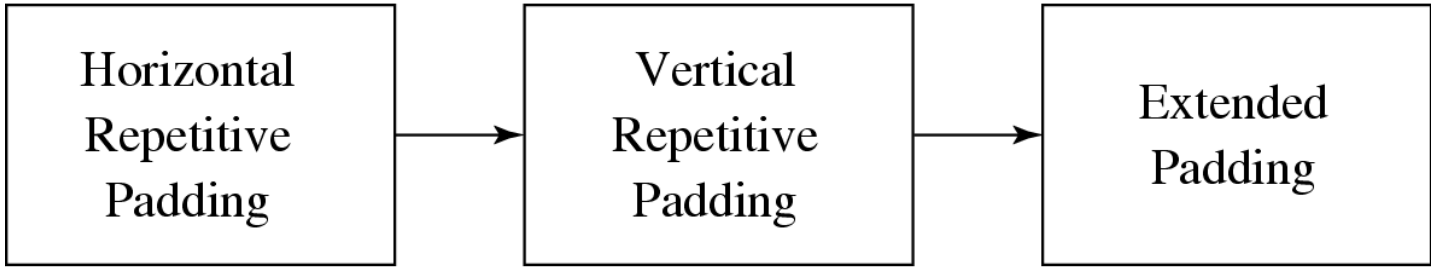



**Fig. 11.16:** A Sequence of Paddings for Reference VOPs in MPEG-4.

- Afterwards, for all **Exterior Macroblocks** that are outside of the VOP but adjacent to one or more Boundary MBs, *extended padding* will be applied.

# Algorithm 11.1 Horizontal Repetitive Padding:

```
BEGIN
   FOR all rows in Boundary MBs in the Reference VOP
     IF ∃ (boundary pixel) in the row
        FOR all interval outside of VOP
              IF interval is bounded by only one boundary
              pixel b
                assign the value of b to all  pixels in
                interval
              ELSE // interval is bounded by two boundary
              pixels b1 and b2
                assign the value of (b1 +b2)/2 to all
                pixels in interval
END
```

- The subsequent Vertical Repetitive Padding algorithm works in a similar manner.

# Example 11.1: Repetitive Paddings

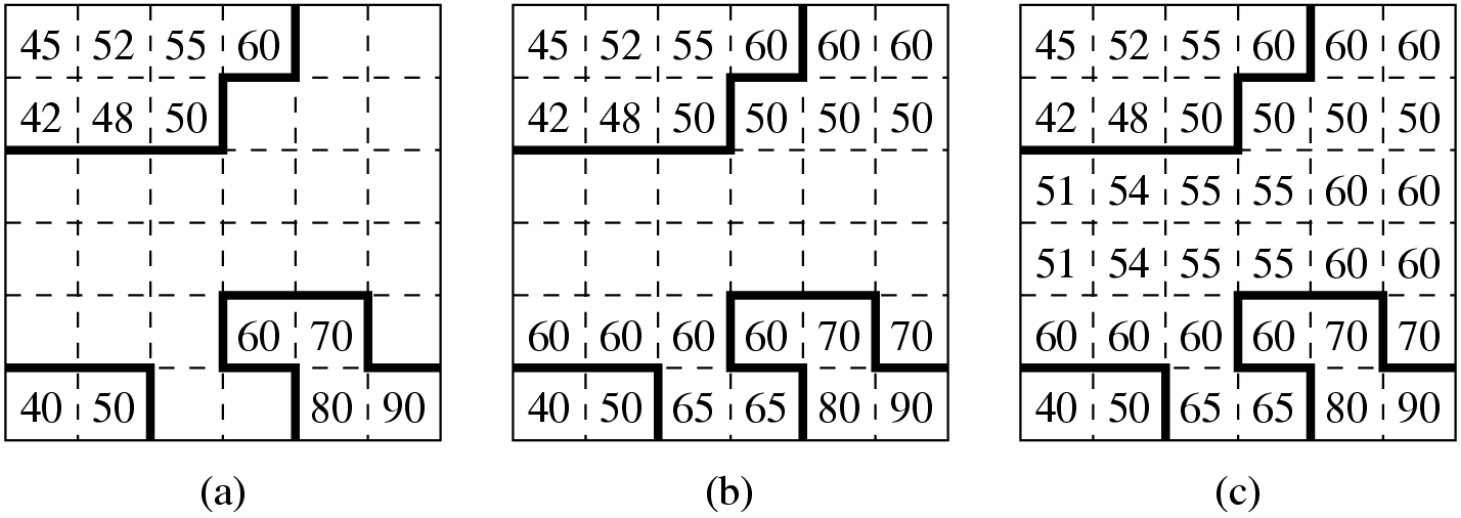



**Fig. 11.17:** An example of Repetitive Padding in a boundary macroblock of a Reference VOP:

(a) Original pixels within the VOP

(b) After Horizontal Repetitive Padding

(c) Followed by Vertical Repetitive Padding.

# Motion Vector Coding

- Let $C(x + k, y + l)$ be pixels of the MB in Target VOP, and $R(x+i+k, y+j+l)$ be pixels of the MB in Reference VOP.
- A **Sum of Absolute Difference (SAD)** for measuring the difference between the two MBs can be defined as:

$$SAD(i,j) = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\left|C(x+k,y+l)-R(x+i+k,y+j+l)\right|\cdot Map(x+k,y+l)$$

$N$ — the size of the MB. $Map(p, q)$ = 1 when $C(p, q)$ is a pixel within the target VOP, otherwise $Map(p, q)$ = 0.

- The vector $(i, j)$ that yields the minimum SAD is adopted as the motion vector $\mathrm{MV}(u, v)$:

$$(u,v)=\{(i,j)\,|\,SAD(i,j)\text{ is minimum},\ i\in[-p,p], j\in[-p,p]\} \qquad (11.3)$$

$p$ — the maximal allowable magnitude for $u$ and $v$.

# Texture Coding

- Texture coding in MPEG-4 can be based on:
  - DCT or
  - Shape Adaptive DCT (SA-DCT).

## I. Texture coding based on DCT

- In I-VOP, the gray values of the pixels in each MB of the VOP are directly coded using the DCT followed by VLC, similar to what is done in JPEG.
- In P-VOP or B-VOP, MC-based coding is employed — it is the prediction error that is sent to DCT and VLC.

- Coding for the Interior MBs:

  - Each MB is 16 × 16 in the luminance VOP and 8 × 8 in the chrominance VOP.

  - Prediction errors from the six 8×8 blocks of each MB are obtained after the conventional motion estimation step.

- Coding for Boundary MBs:

  - For portions of the Boundary MBs in the Target VOP outside of the VOP, zeros are padded to the block sent to DCT since ideally prediction errors would be near zero inside the VOP.

  - After MC, texture prediction errors within the Target VOP are obtained.

# II. SA-DCT based coding for Boundary MBs

- Shape Adaptive DCT (SA-DCT) is another texture coding method for boundary MBs.
- Due to its effectiveness, SA-DCT has been adopted for coding boundary MBs in MPEG-4 Version 2.
- It uses the 1D DCT-N transform and its inverse, IDCT-N:

– **1D DCT-N**:

$$F(u) = \sqrt{\frac{2}{N}} C(u) \sum_{i=0}^{N-1} \cos \frac{(2i+1)u\pi}{2N} f(i) \qquad (12.2)$$

– **1D IDCT-N**:

$$\tilde{f}(i) = \sum_{u=0}^{N-1} \sqrt{\frac{2}{N}} C(u) \cos \frac{(2i+1)u\pi}{2N} F(u) \qquad (12.3)$$

where $i = 0, 1, \ldots, N-1$; $u = 0, 1, \ldots, N-1$ and

$$C(u) = \begin{cases} \frac{\sqrt{2}}{2} & if \quad u = 0, \\ 1 & otherwise. \end{cases}$$

- SA-DCT is a 2D DCT and it is computed as a separable 2D transform in two iterations of 1D DCT-N.
- Fig. 11.18 illustrates the process of texture coding for boundary MBs using the Shape Adaptive DCT (SA-DCT).

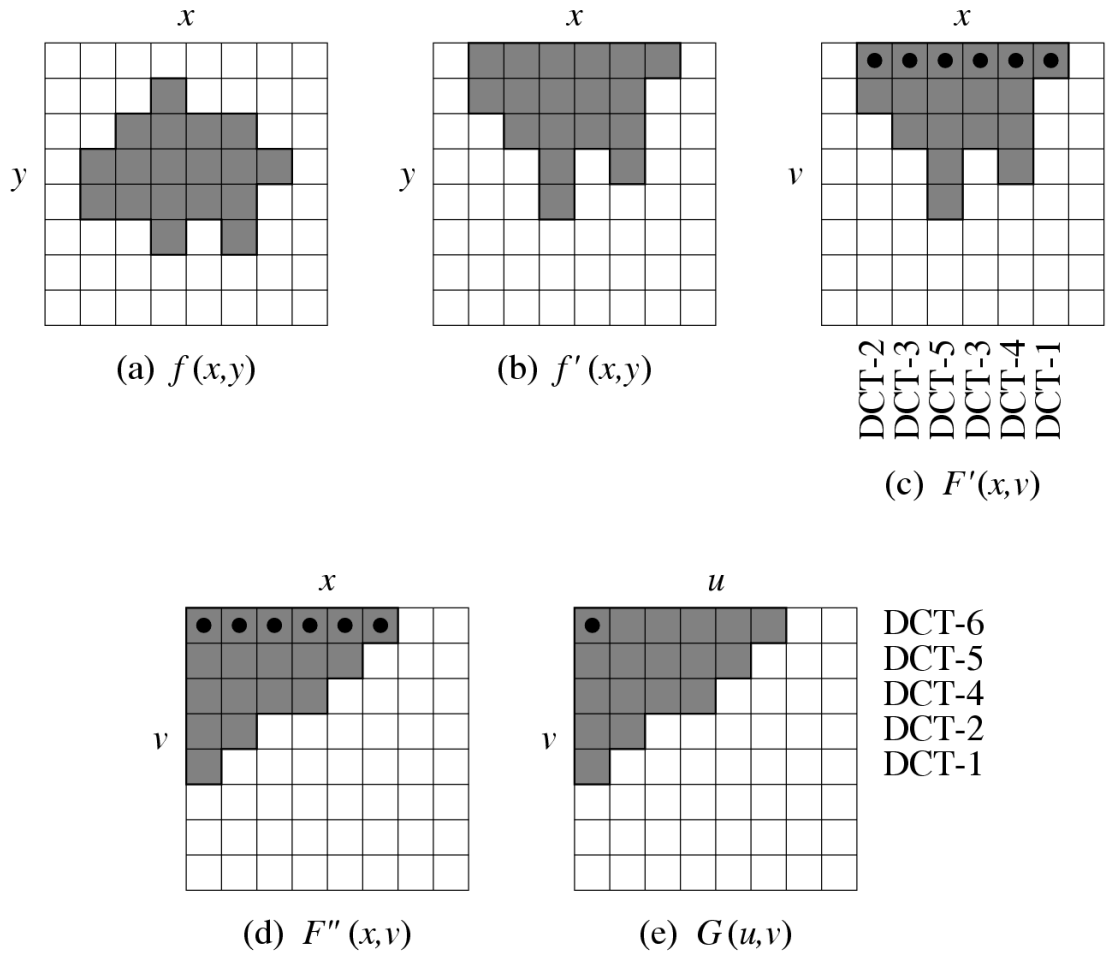



**Fig. 11.18:** Texture Coding for Boundary MBs Using the Shape Adaptive DCT (SA-DCT).

# Shape Coding

- MPEG-4 supports two types of shape information, **binary** and **gray scale**.
- Binary shape information can be in the form of a binary map (also known as *binary alpha map*) that is of the size as the rectangular bounding box of the VOP.
- A value '1' (opaque) or '0' (transparent) in the bitmap indicates whether the pixel is inside or outside the VOP.
- Alternatively, the gray-scale shape information actually refers to the *transparency* of the shape, with gray values ranging from 0 (completely transparent) to 255 (opaque).

# Binary Shape Coding

- *BABs* (*Binary Alpha Blocks*): to encode the binary alpha map more efficiently, the map is divided into 16×16. blocks

- It is the boundary BABs that contain the contour and hence the shape information for the VOP — the subject of binary shape coding.

- Two bitmap-based algorithms:
  (a) **Modified Modified READ (MMR)**.
  (b) **Context-based Arithmetic Encoding (CAE)**.

# Modified Modified READ (MMR)

- MMR is basically a series of simplifications of the **Relative Element Address Designate** (READ) algorithm

- The READ algorithm starts by identifying five pixel locations in the previous and current lines:
  - $a_0$: the last pixel value known to both the encoder and decoder;
  - $a_1$: the transition pixel to the right of $a_0$;
  - $a_2$: the second transition pixel to the right of $a_0$;
  - $b_1$: the first transition pixel whose color is opposite to $a_0$ in the previously coded line; and
  - $b_2$: the first transition pixel to the right of $b_1$ on the previously coded line.

# Modified Modified READ (MMR) (Cont'd)

- The READ algorithm works by examining the relative position of these pixels:
  - At any point in time, both the encoder and decoder know the position of $a_0$, $b_1$, and $b_2$ while the positions $a_1$ and $a_2$ are known only in the encoder.
  - Three coding modes are used:
    1. If the run lengths on the previous line and the current line are similar, the distance between $a_1$ and $b_1$ should be much smaller than the distance between $a_0$ and $a_1$. The *vertical mode* encodes the current run length as $a_1 - b_1$.
    2. If the previous line has no similar run length, the current run length is coded using one- dimensional run length coding — *horizontal mode*.
    3. If $a_0 \leq b_1 < b_2 < a_1$, simply transmit a codeword indicating it is in pass mode and advance $a_0$ to the position under $b_2$ and continue the coding process.

- Some simplifications can be made to the READ algorithm for practical implementation.
  - For example, if $\|a_1 - b_1\| < 3$, then it is enough to indicate that we can apply the vertical mode.
  - Also, to prevent error propagation, a $k$-factor is defined such that every $k$ lines must contain at least one line coded using conventional run length coding.
  - These modifications constitute the *Modified READ* algorithm used in the G3 standard. The MMR (Modified Modified READ) algorithm simply removes the restrictions imposed by the $k$-factor.

# CAE (Context-based Arithmetic Encoding)

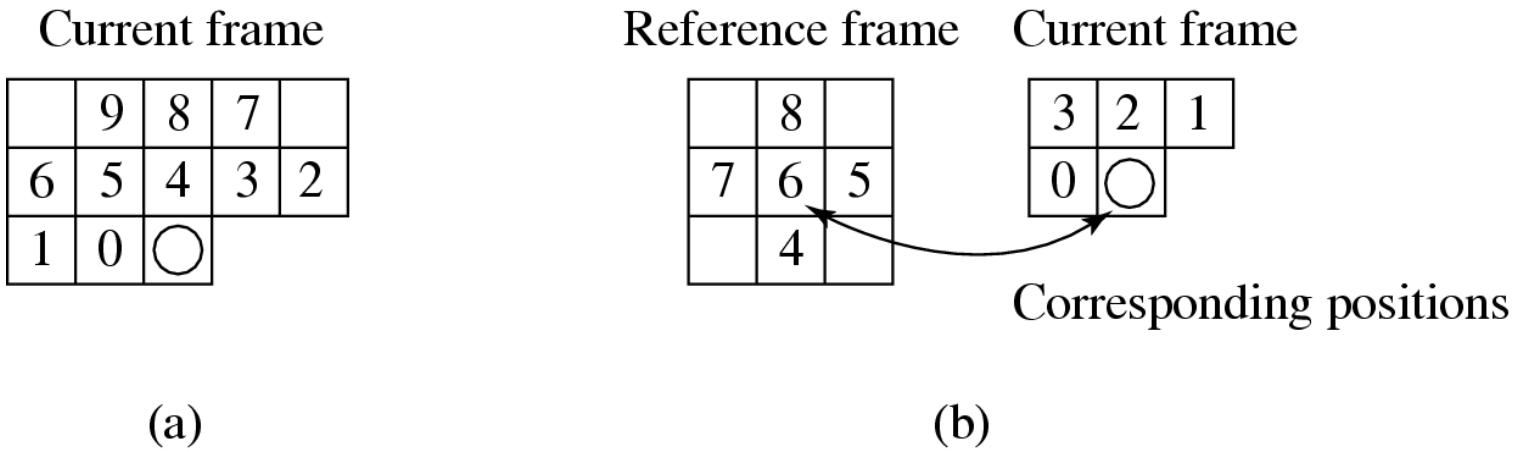



**Fig. 11.19:** Contexts in CAE for a pixel in the boundary BAB.
(a) Intra-CAE, (b) Inter-CAE.

# CAE (con't)

- Certain contexts (e.g., all 1s or all 0s) appear more frequently than others.

  With some prior statistics, a probability table can be built to indicate the probability of occurrence for each of the $2^k$ contexts, where $k$ is the number of neighboring pixels.

- Each pixel can look up the table to find a probability value for its context. CAE simply scans the 16 × 16 pixels in each BAB sequentially and applies Arithmetic coding to eventually derive a single floating-point number for the BAB.

- Inter-CAE mode is a natural extension of intra-CAE: it involves both the target and reference alpha maps.

# Gray-scale Shape Coding

- The **gray-scale** here is used to describe the **transparency** of the shape, not the texture.
- **Gray-scale shape coding** in MPEG-4 employs the same technique as in the texture coding described above.
  - Uses the alpha map and block-based motion compensation, and encodes the prediction errors by DCT.
  - The boundary MBs need padding as before since not all pixels are in the VOP.

# Static Texture Coding

- MPEG-4 uses wavelet coding for the texture of static objects.
- The coding of subbands in MPEG-4 static texture coding is conducted in the following manner:
  - The subbands with the lowest frequency are coded using DPCM. Prediction of each coefficient is based on three neighbors.
  - Coding of other subbands is based on a multiscale zero-tree wavelet coding method.
- The multiscale zero-tree has a Parent-Child Relation tree (PCR tree) for each coefficient in the lowest frequency sub-band to better track locations of all coefficients.
- The degree of quantization also affects the data rate.

# Sprite Coding

- A **sprite** is a graphic image that can freely move around within a larger graphic image or a set of images.
- To separate the foreground object from the background, we introduce the notion of a **sprite panorama**: a still image that describes the static background over a sequence of video frames.
  - The large sprite panoramic image can be encoded and sent to the decoder only once at the beginning of the video sequence.
  - When the decoder receives separately coded foreground objects and parameters describing the camera movements thus far, it can reconstruct the scene in an efficient manner.
  - Fig. 12.10 shows a sprite which is a panoramic image stitched from a sequence of video frames.

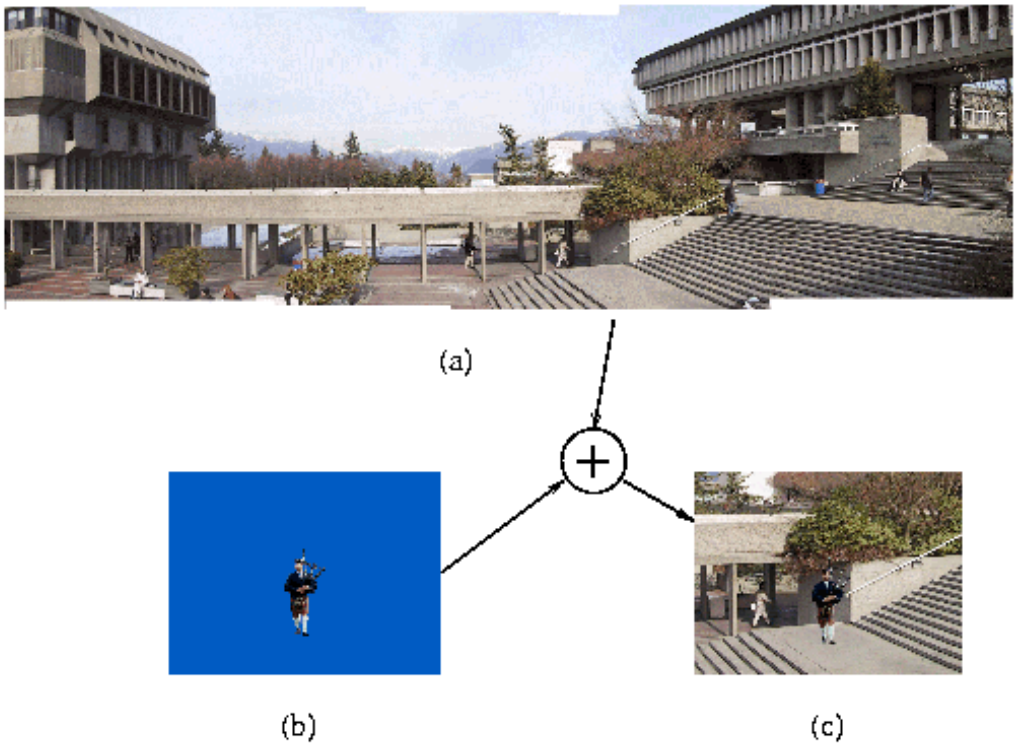



**Fig. 11.20:** Sprite Coding. (a) The sprite panoramic image of the background, (b) the foreground object (piper) in a blue-screen image, (c) the composed video scene.

*Piper image courtesy of Simon Fraser University Pipe Band.*

# Global Motion Compensation (GMC)

- "Global" – overall change due to camera motions (pan, tilt, rotation and zoom)

  Without GMC this will cause a large number of significant motion vectors

- There are four major components within the GMC algorithm:
  - Global motion estimation
  - Warping and blending
  - Motion trajectory coding
  - Choice of LMC (Local Motion Compensation) or GMC.

- Global motion is computed by minimizing the sum of square differences between the sprite $S$ and the global motion compensated image $I'$:

$$E = \sum_{i=1}^{N} (S(x_i, y_i) - I'(x_i', y_i'))^2 \tag{12.4}$$

- The motion over the whole image is then parameterized by a perspective motion model using eight parameters defined as:

$$x_i' = \frac{a_0 + a_1 x_i + a_2 y_i}{a_6 x_i + a_7 y_i + 1},$$

$$y_i' = \frac{a_3 + a_4 x_i + a_5 y_i}{a_6 x_i + a_7 y_i + 1} \tag{12.5}$$

# 11.4.3 Synthetic Object Coding in MPEG-4

## 2D Mesh Object Coding

- **2D mesh**: a tessellation (or partition) of a 2D planar region using polygonal patches:
  - The vertices of the polygons are referred to as *nodes* of the mesh.
  - The most popular meshes are *triangular meshes* where all polygons are triangles.
  - The MPEG-4 standard makes use of two types of 2D mesh: **uniform mesh** and **Delaunay mesh**
  - 2D mesh object coding is compact. All coordinate values of the mesh are coded in half-pixel precision.
  - Each 2D mesh is treated as a *mesh object plane* (*MOP*).

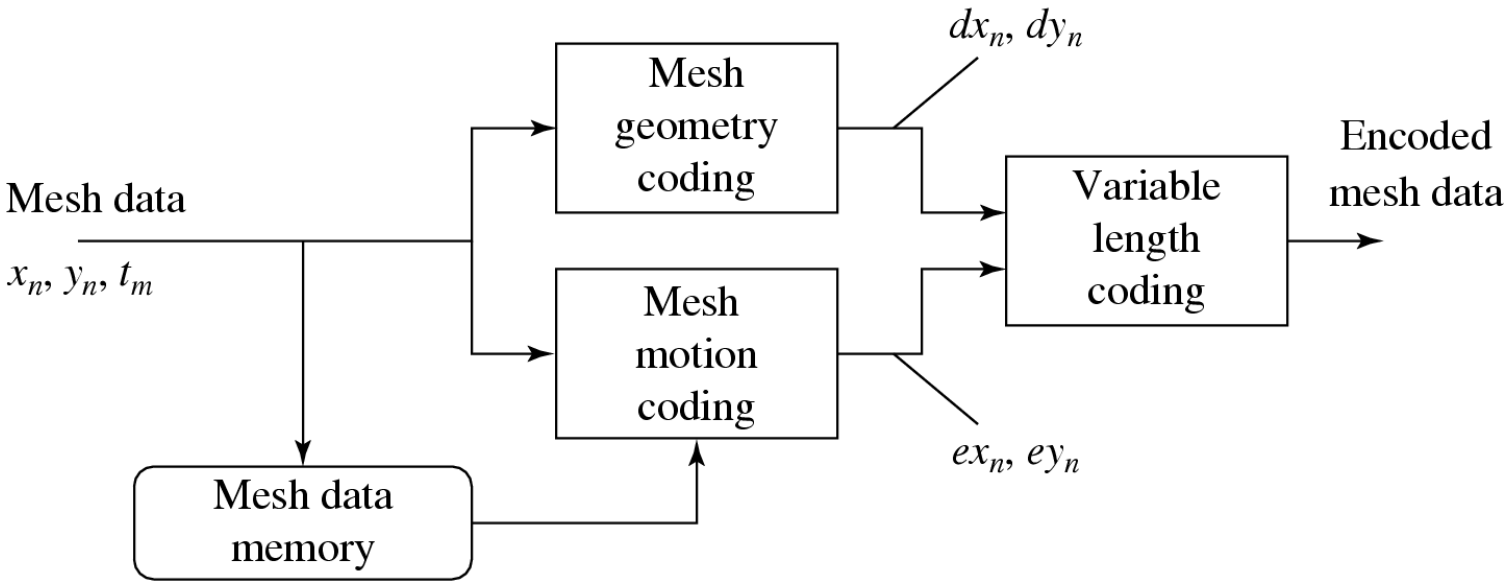



**Fig. 11.21**: 2D Mesh Object Plane (MOP) Encoding Process

# 2D Mesh Geometry Coding

- MPEG-4 allows four types of uniform meshes with different triangulation structures.

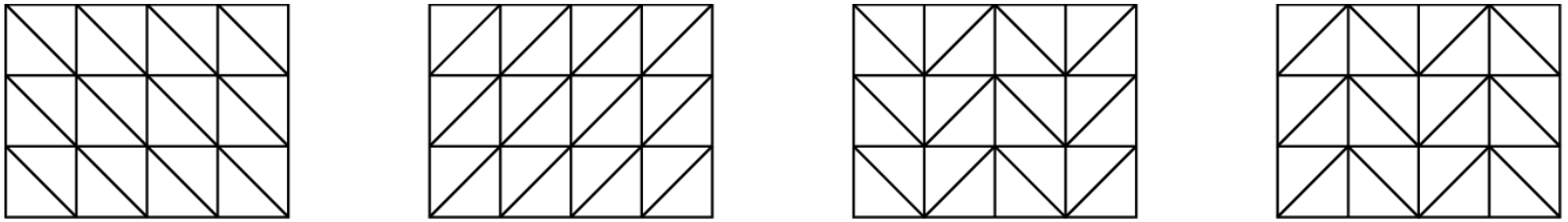



**Fig. 11.22:** Four Types of Uniform Meshes.

- **Definition**: If $D$ is a Delaunay triangulation, then any of its triangles $t_n = (P_i, P_j, P_k) \in D$ satisfies the property that the circumcircle of $t_n$ does not contain in its interior any other node point $P_l$.

- A Delaunay mesh for a video object can be obtained in the following steps:

  1. **Select boundary nodes of the mesh**: A polygon is used to approximate the boundary of the object.

  2. **Choose interior nodes**: Feature points, e.g., edge points or corners, within the object boundary can be chosen as interior nodes for the meh.

  3. **Perform Delaunay triangulation**: A *constrained Delaunay triangulation* is performed on the boundary and interior nodes with the polygonal boundary used as a constraint.

# Constrained Delaunay Triangulation

- Interior edges are first added to form new triangles.
- The algorithm will examine each interior edge to make sure it is *locally Delaunay*.
- Given two triangles ($P_i$, $P_j$, $P_k$) and ($P_j$, $P_k$, $P_l$) sharing an edge $\overline{jk}$, if ($P_i$, $P_j$, $P_k$) contains $P_l$ or ($P_j$, $P_k$, $P_l$) contains $P_i$ in its interior, then $\overline{jk}$ is not locally Delaunay, and it will be replaced by a new edge $\overline{il}$ .
- If $P_l$ falls exactly on the circumcircle of ($P_i$, $P_j$, $P_k$) (and accordingly, $P_i$ also falls exactly on the circumcircle of ($P_j$, $P_k$, $P_l$)), then $\overline{jk}$ will be viewed as locally Delaunay only if $P_i$ or $P_l$ has the largest x coordinate among the four nodes.

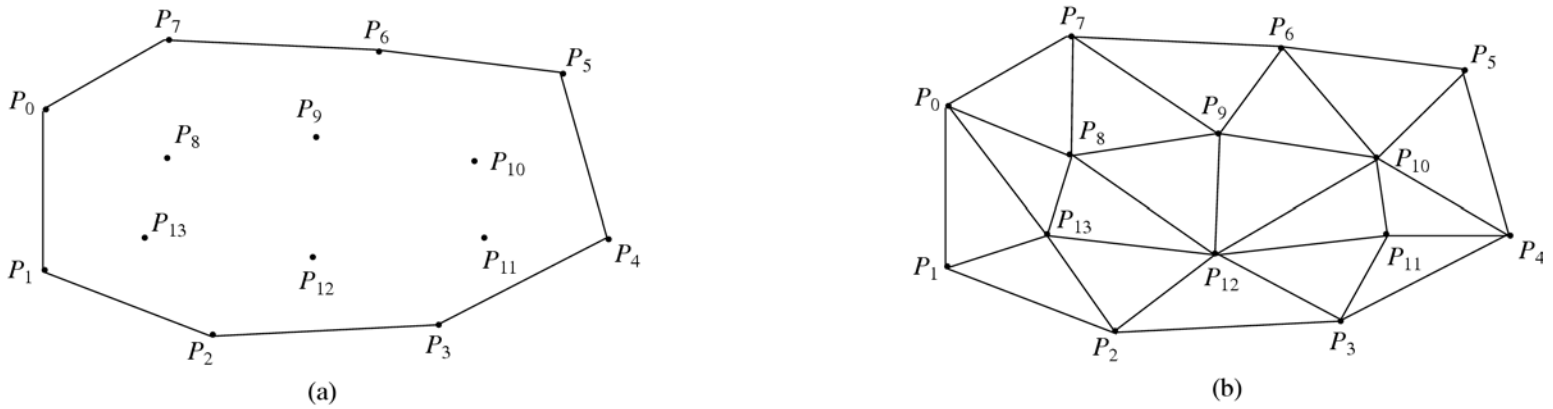



**Fig. 11.23:** Delaunay Mesh: (a) Boundary nodes ($P_0$ to $P_7$) and Interior nodes ($P_8$ to $P_{13}$). (b) Triangular mesh obtained by Constrained Delaunay Triangulation.

- Except for the first location $(x_0, y_0)$, all subsequent coordinates are coded differentially — that is, for $n \geq 1$,

$$dx_n = x_n - x_n{-}1,\ dy_n = y_n - y_n{-}1, \qquad (12.6)$$

and afterward, $dx_n$, $dy_n$ are variable-length coded.

# 2D Mesh Motion Coding

- A new mesh structure can be created only in the Intra-frame, and its triangular topology will not alter in the subsequent Inter-frames — enforces a one-to-one mapping in 2D mesh motion estimation.

- For any MOP triangle ($P_i$, $P_j$, $P_k$), if the motion vectors for $P_i$ and $P_j$ are known to be $\mathrm{MV_i}$ and $\mathrm{MV_j}$, then a prediction $\mathrm{Pred_k}$ will be made for the motion vector of $P_k$ and this is rounded to a half-pixel precision:

$$\mathrm{Pred_k} = 0.5 \cdot (\mathrm{MV_i} + \mathrm{MV_j}) \tag{12.7}$$

The prediction error $\mathrm{e_k}$ is coded as

$$\mathrm{e_k} = \mathrm{MV_k} - \mathrm{Pred_k}. \tag{12.8}$$

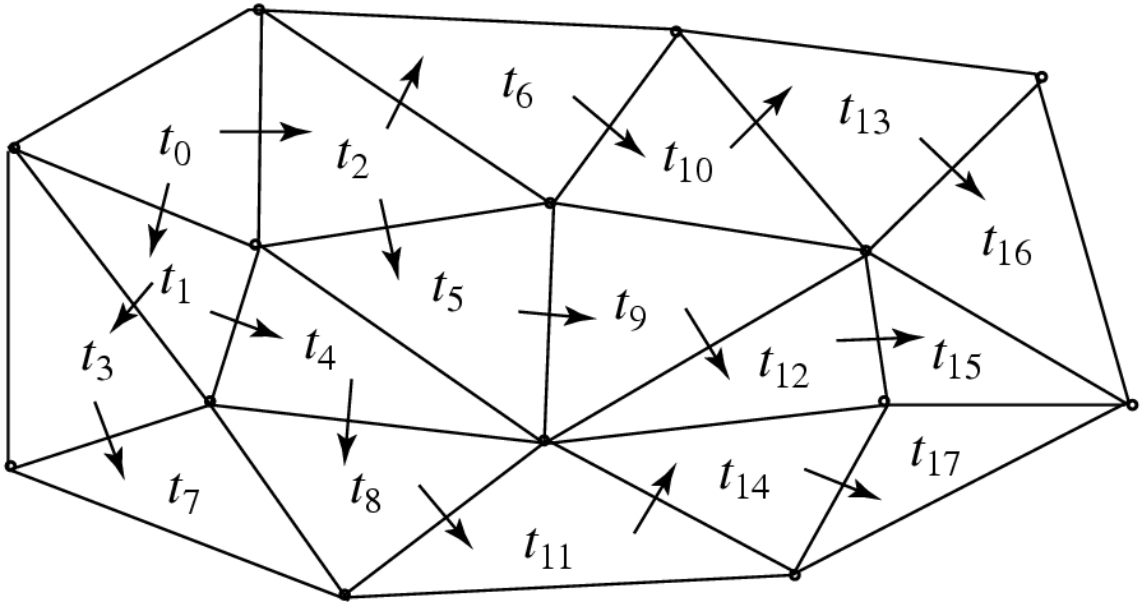



**Fig. 11.24:** A breadth-first order of MOP triangles for 2D mesh motion coding.

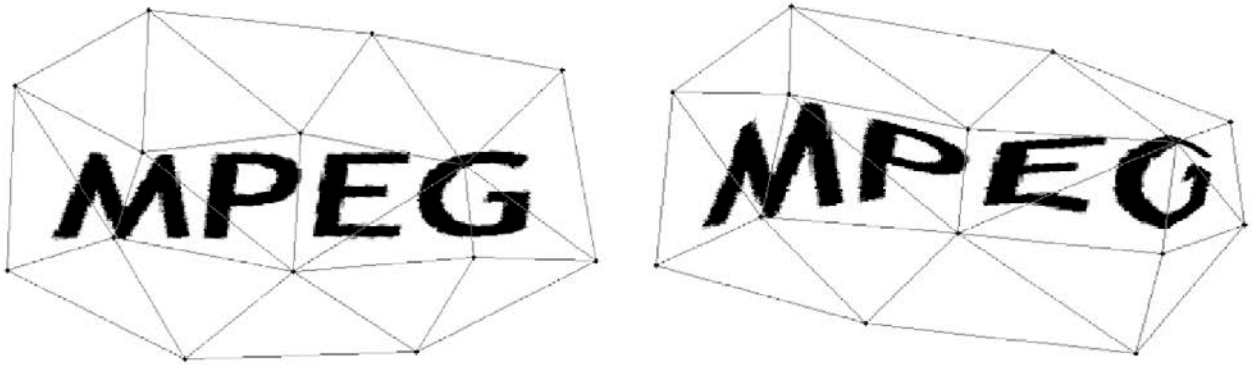



**Fig. 11.25:** Mesh-based texture mapping for 2D object animation.

# 3D Model-Based Coding

- MPEG-4 has defined special 3D models for **face objects** and **body objects** because of the frequent appearances of human faces and bodies in videos.
- Some of the potential applications for these new video objects include teleconferencing, human-computer interfaces, games, and e-commerce.
- MPEG-4 goes beyond wireframes so that the surfaces of the face or body objects can be shaded or texture-mapped.

# Face Object Coding and Animation

- MPEG-4 has adopted a generic default face model, which was developed by VRML Consortium.
- **Face Animation Parameters (FAPs)** can be specified to achieve desirable animations — deviations from the original "neutral" face.
- In addition, **Face Definition Parameters (FDPs)** can be specified to better describe individual faces.
- Fig. 11.26 shows the feature points for FDPs. Feature points that can be affected by animation (FAPs) are shown as solid circles, and those that are not affected are shown as empty circles.

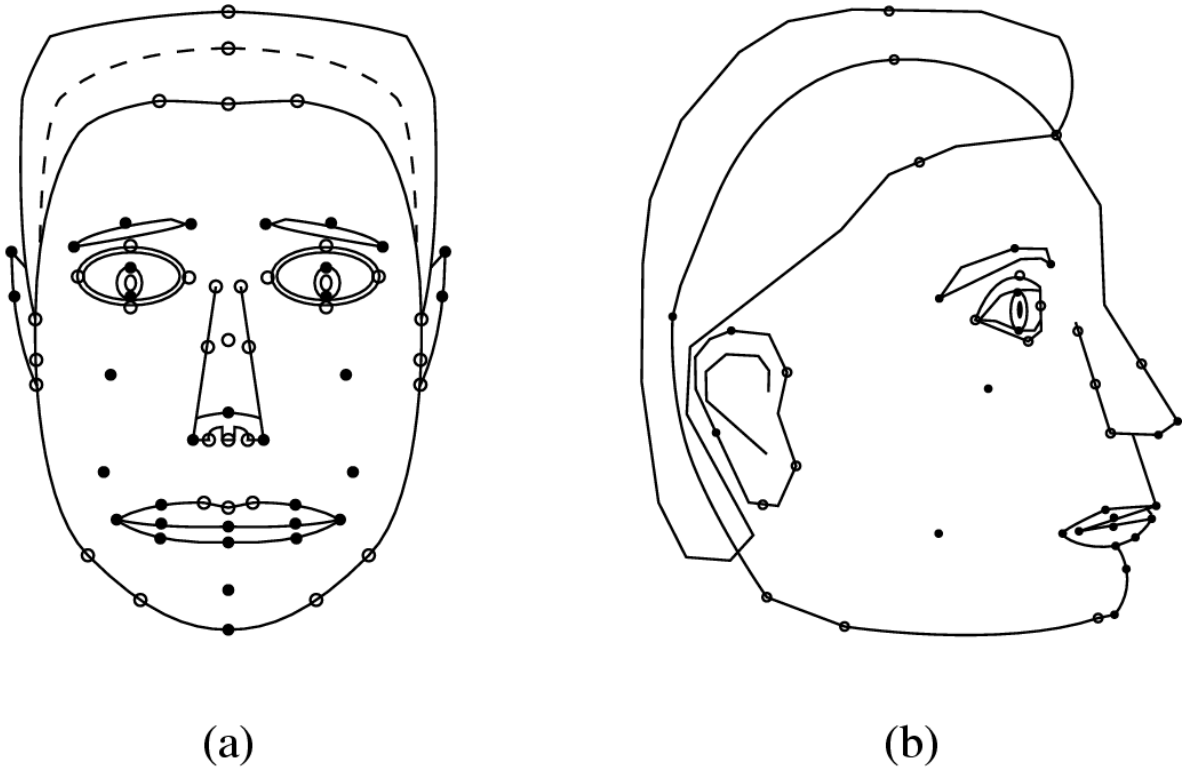



**Fig. 11.26:** Feature Points for Face Definition Parameters (FDPs). (Feature points for teeth and tongue not shown.)

# Body Object Coding and Animation

- MPEG-4 Version 2 introduced **body objects**, which are a natural extension to face objects.

- Working with the Humanoid Animation (H-Anim) Group in the VRML Consortium, a generic virtual human body with default posture is adopted.

  - The default posture is a standing posture with feet pointing to the front, arms on the side and palms facing inward.

  - There are 296 **Body Animation Parameters (BAPs)**. When applied to any MPEG-4 compliant generic body, they will produce the same animation.

- A large number of BAPs are used to describe joint angles connecting different body parts: spine, shoulder, clavicle, elbow, wrist, finger, hip, knee, ankle, and toe — yields 186 degrees of freedom to the body, and 25 degrees of freedom to each hand alone.

- Some body movements can be specified in multiple levels of detail.

- For specific bodies, **Body Definition Parameters (BDPs)** can be specified for body dimensions, body surface geometry, and optionally, texture.

- The coding of BAPs is similar to that of FAPs: quantization and predictive coding are used, and the prediction errors are further compressed by arithmetic coding.

# 11.4.4 MPEG-4 Parts, Profiles and Levels

- The standardization of Profiles and Levels in MPEG-4 serve two main purposes:

  (a) ensuring interoperability between implementations

  (b) allowing testing of conformance to the standard

- MPEG-4 not only specified Visual profiles and Audio profiles, but it also specified Graphics profiles, Scene description profiles, and one Object descriptor profile in its Systems part.

# 11.5 MPEG-7

- The main objective of MPEG-7 is to serve the need of audio-visual content-based retrieval (or audiovisual object retrieval) in applications such as digital libraries.

- Nevertheless, it is also applicable to any multimedia applications involving the generation (*content creation*) and usage (*content consumption*) of multimedia data.

- MPEG-7 became an International Standard in September 2001 — with the formal name **Multimedia Content Description Interface**.

# Applications Supported by MPEG-7

- MPEG-7 supports a variety of multimedia applications. Its data may include still pictures, graphics, 3D models, audio, speech, video, and composition information (how to combine these elements).
- These MPEG-7 data elements can be represented in textual format, or binary format, or both.
- Fig. 11.27 illustrates some possible applications that will benefit from the MPEG-7 standard.

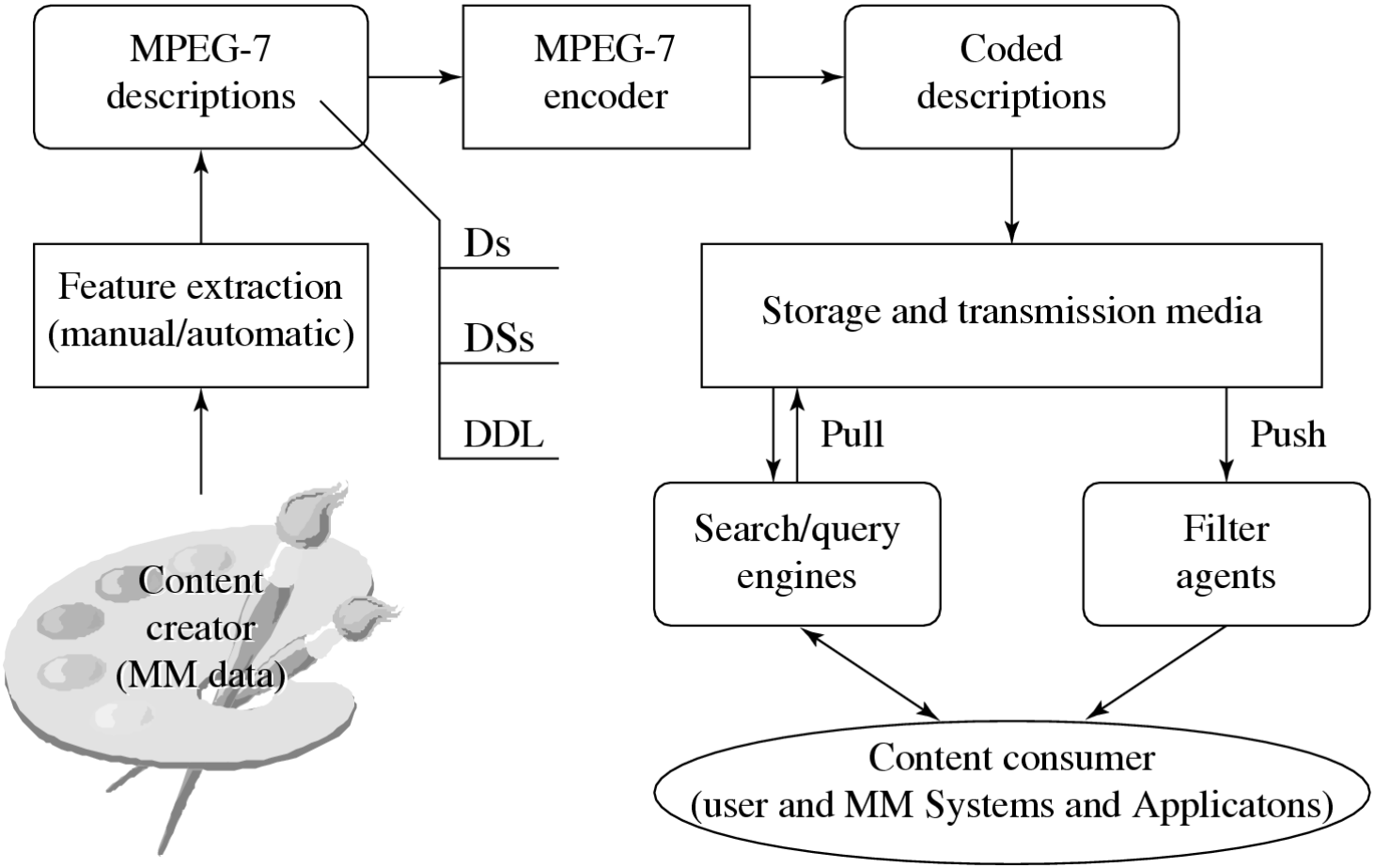



**Fig. 11.27:** Possible Applications using MPEG-7.

# MPEG-7 and Multimedia Content Description

- MPEG-7 has developed Descriptors **(D)**, Description Schemes **(DS)** and Description Definition Language **(DDL)**. The following are some of the important terms:
  - **Feature** — characteristic of the data.
  - **Description** — a set of instantiated Ds and DSs that describes the structural and conceptual information of the content, the storage and usage of the content, etc.
  - **D** — definition (syntax and semantics) of the feature.
  - **DS** — specification of the structure and relationship between Ds and between DSs.
  - **DDL** — syntactic rules to express and combine DSs and Ds.

- The scope of MPEG-7 is to standardize the Ds, DSs and DDL for descriptions. The mechanism and process of producing and consuming the descriptions are beyond the scope of MPEG-7.

# Descriptor (D)

- The descriptors are chosen based on a comparison of their performance, efficiency, and size. Low-level visual descriptors for basic visual features include:
  - **Color**
    * Color space. (a) RGB, (b) YCbCr, (c) HSV (hue, saturation, value), (d) HMMD (HueMaxMinDiff), (e) 3D color space derivable by a 3 × 3 matrix from RGB, (f) monochrome.
    * Color quantization. (a) Linear, (b) nonlinear, (c) lookup tables.
    * Dominant colors.
    * Scalable color.
    * Color layout.
    * Color structure.
    * Group of Frames/Group of Pictures (GoF/GoP) color.

- **Texture**
  - Homogeneous texture.
  - Texture browsing.
  - Edge histogram.

- **Shape**
  - Region-based shape.
  - Contour-based shape.
  - 3D shape.

- **Motion**
  - Camera motion (see Fig. 11.28).
  - Object motion trajectory.
  - Parametric object motion.
  - Motion activity.

- **Localization**
  - Region locator.
  - Spatiotemporal locator.

- **Others**
  - Face recognition.

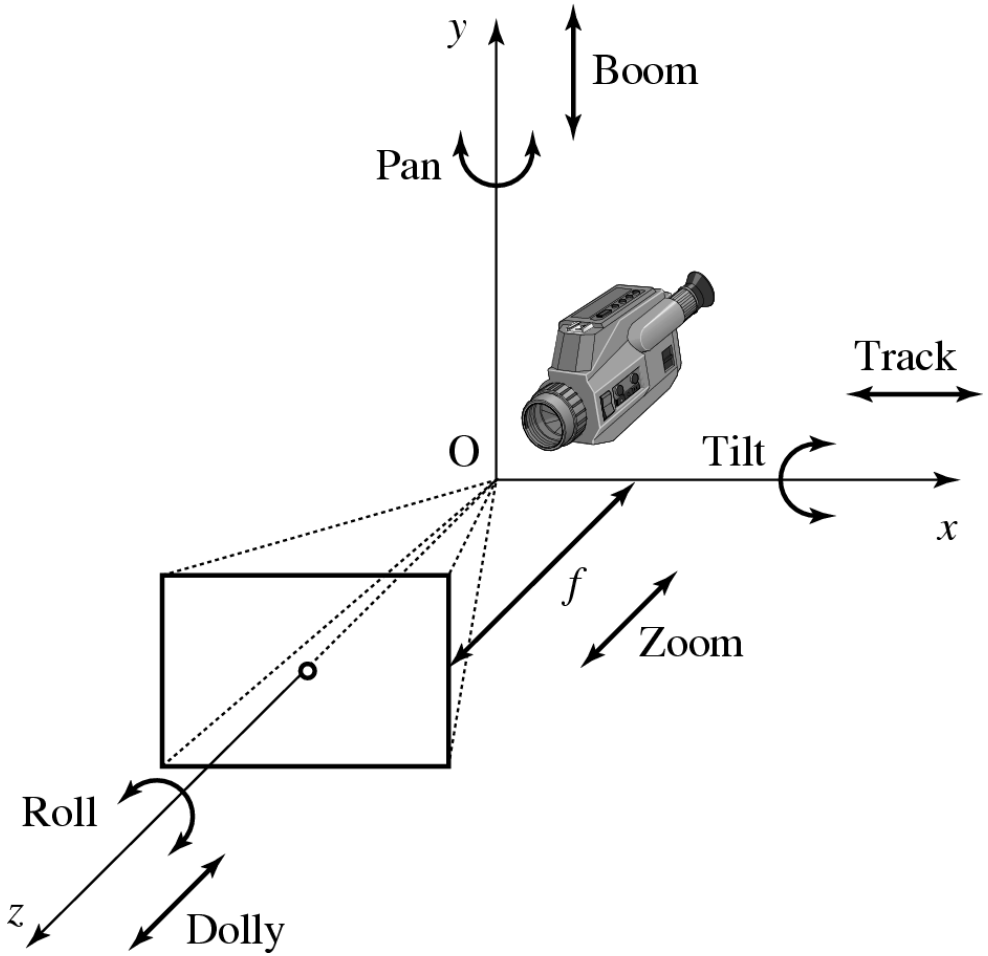



**Fig. 11.28:** Camera motions: pan, tilt, roll, dolly, track, and boom.

# Description Scheme (DS)

- **Basic elements**
  - Datatypes and mathematical structures.
  - Constructs.
  - Schema tools.

- **Content Management**
  - Media Description.
  - Creation and Production Description.
  - Content Usage Description.

- **Content Description**
  - Structural Description.

A *Segment* DS, for example, can be implemented as a class object. It can have five subclasses: *Audiovisual segment DS*, *Audio segment DS*, *Still region DS*, *Moving region DS*, and *Video segment DS*. The subclass DSs can recursively have their own subclasses.

– Conceptual Description.

- **Navigation and access**
  - Summaries.
  - Partitions and Decompositions.
  - Variations of the Content.
- **Content Organization**
  - Collections.
  - Models.
- **User Interaction**
  - User Preference.

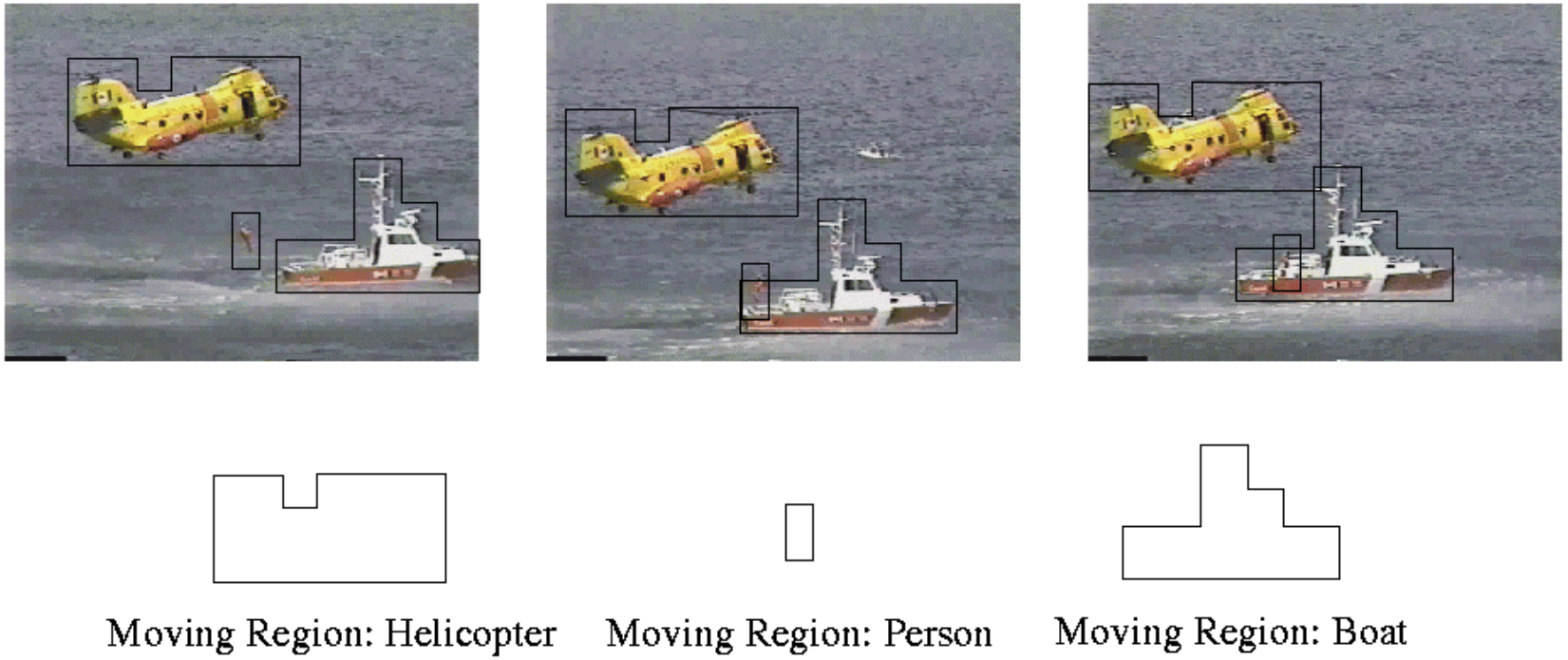



**Fig. 11.29:** MPEG-7 video segment.

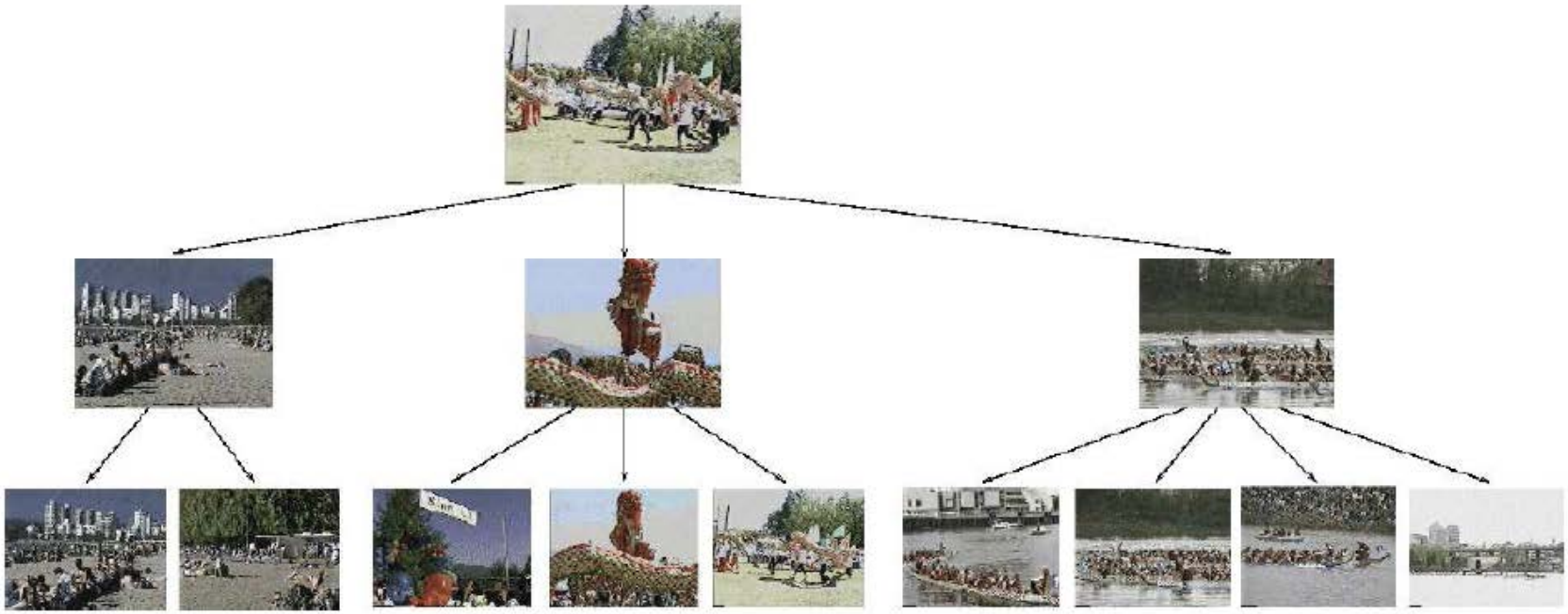


**Fig. 11.30:** A video summary.

# Description Definition Language (DDL)

- MPEG-7 adopted the XML Schema Language initially developed by the WWW Consortium (W3C) as its Description Definition Language (DDL). Since XML Schema Language was not designed specifically for audiovisual contents, some extensions are made to it:
  - Array and matrix data types.
  - Multiple media types, including audio, video, and audiovisual presentations.
  - Enumerated data types for **MimeType**, **CountryCode**, **RegionCode**, **CurrencyCode**, and **CharacterSetCode**.
  - Intellectual Property Management and Protection **(IPMP)** for Ds and DSs.