Chapter 12
MPEG Video Coding II
— MPEG-4, 7 and Beyond

12.1 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 12.1 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.

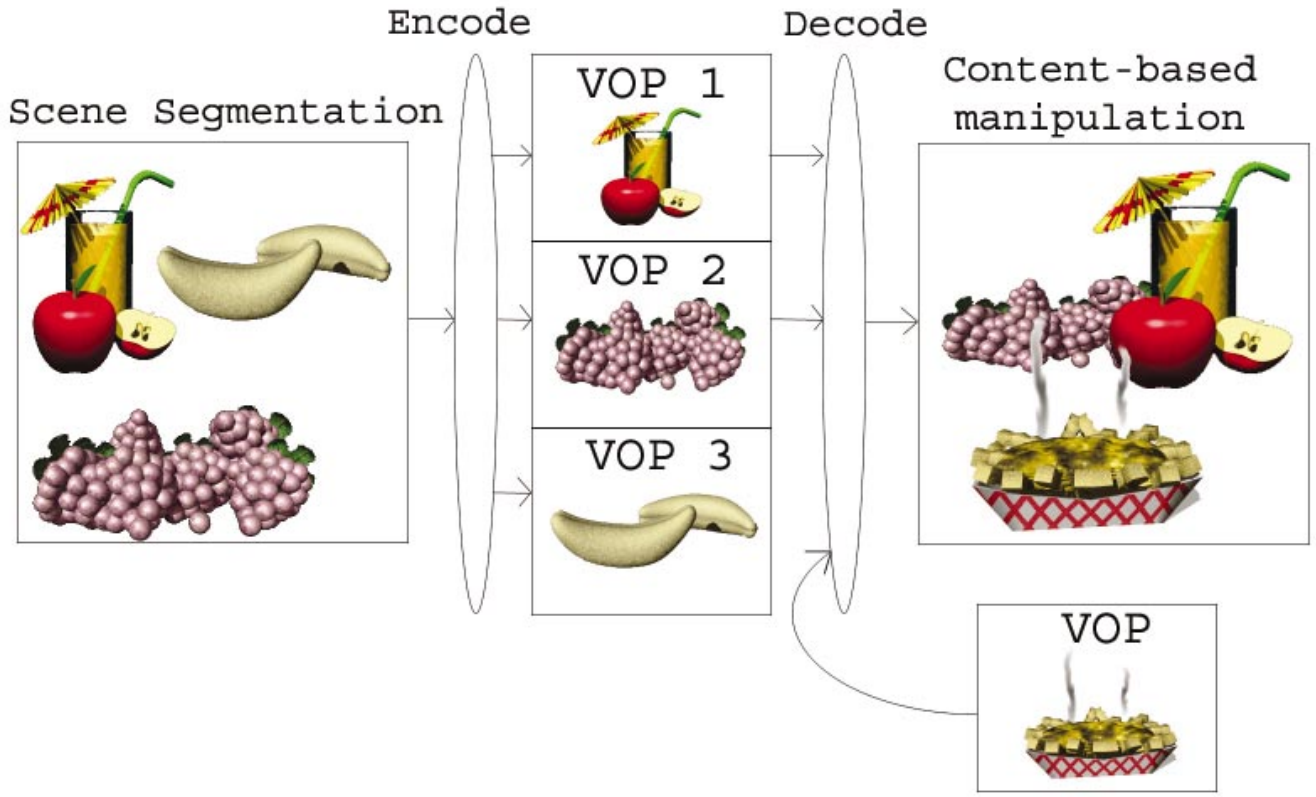


Fig. 12.1: Composition and Manipulation of MPEG-4 Videos.

Overview of MPEG-4 (Cont'd)

- MPEG-4 (Fig. 12.2(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.

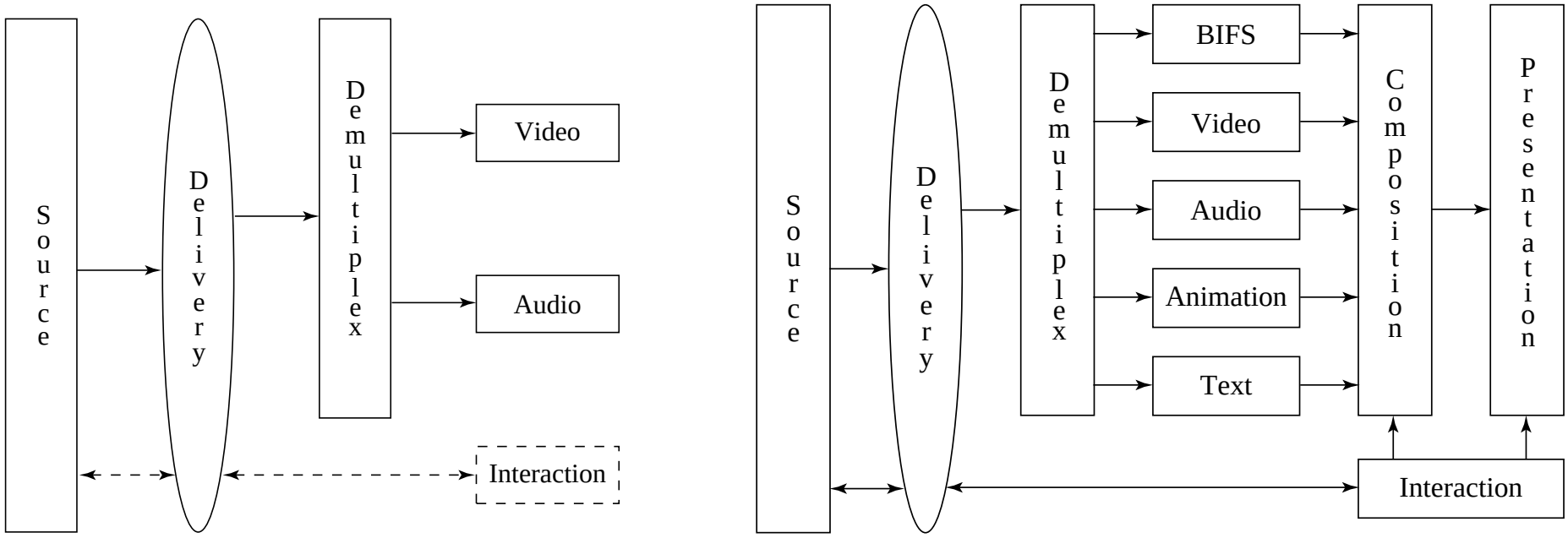


Fig. 12.2: Comparison of interactivities in MPEG standards: (a) reference models in MPEG-1 and 2 (interaction in dashed lines supported only by MPEG-2); (b) MPEG-4 reference model.

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.

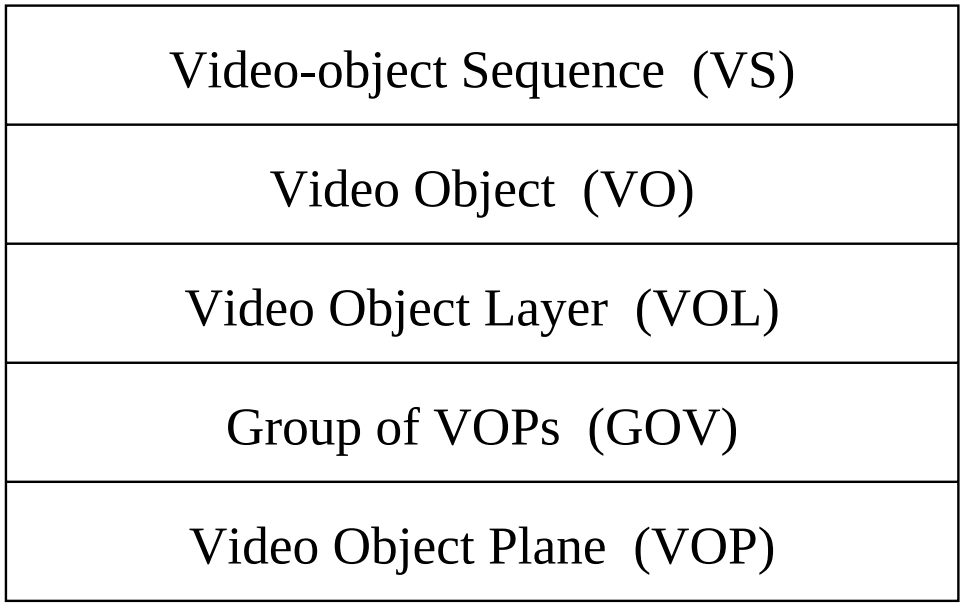


Fig. 12.3: Video Object Oriented Hierarchical Description of a Scene in MPEG-4 Visual Bitstreams.

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.

12.2 Object-based Visual 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. 12.4 (c) illustrates a possible example in which both potential matches yield small prediction errors for block-based coding.

- Fig. 12.4 (d) shows that each VOP is of arbitrary shape and ideally will obtain a unique motion vector consistent with the actual object motion.

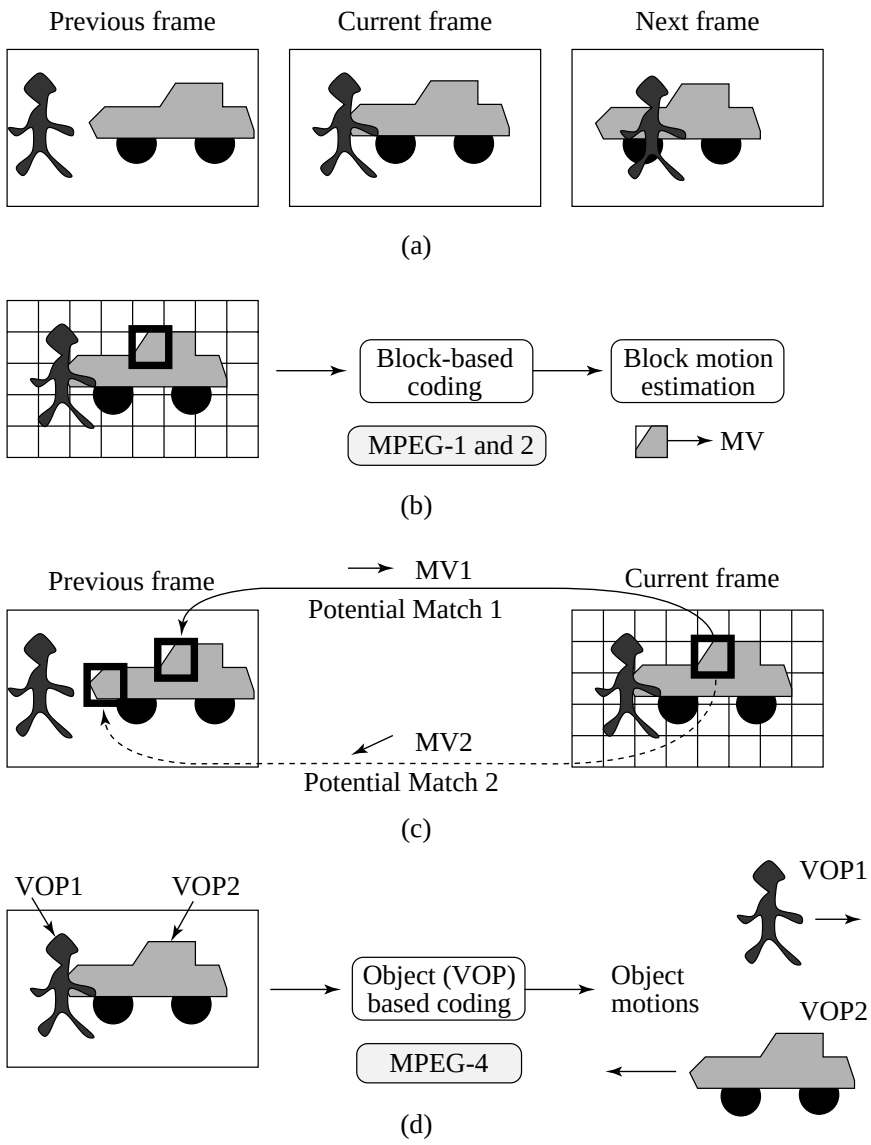


Fig. 12.4: Comparison between Block-based Coding and Object-based Coding.

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-VOP*s if only forward prediction is employed, or *B-VOP*s 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. 12.5 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.

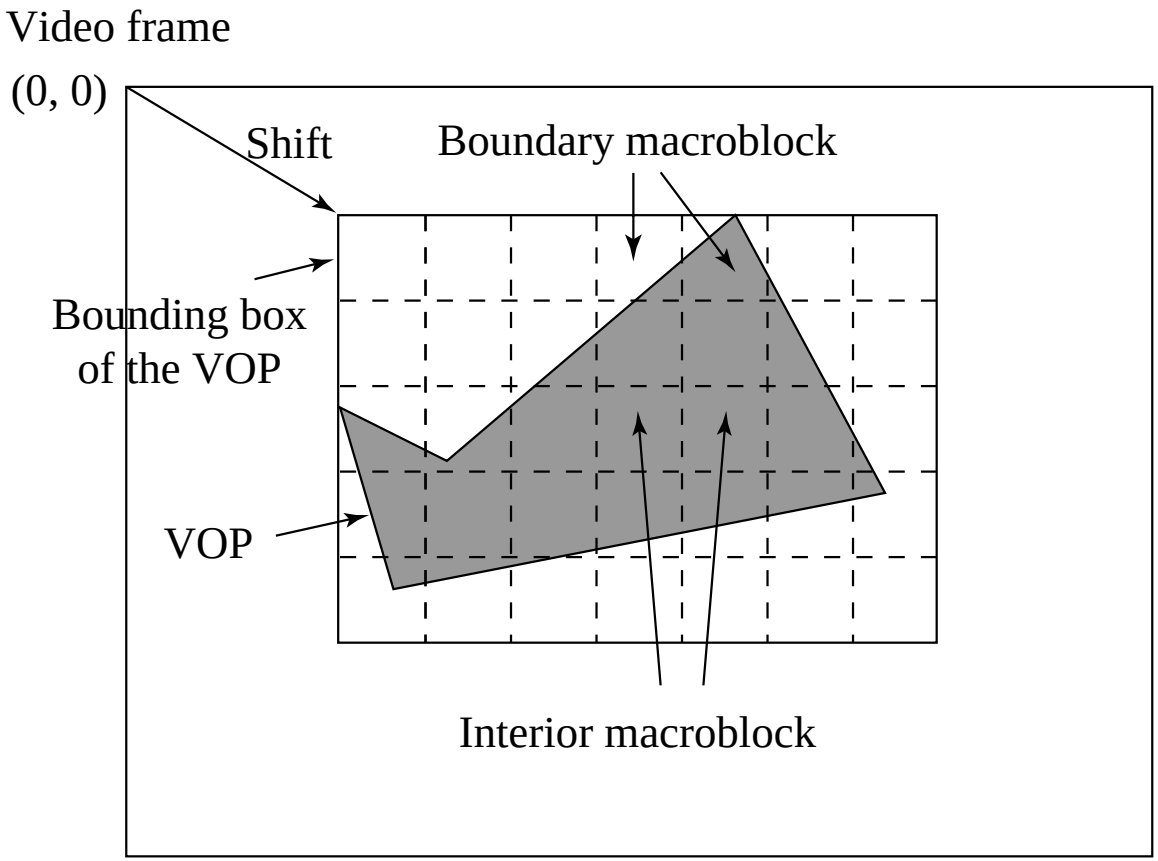


Fig. 12.5: Bounding Box and Boundary Macroblocks of VOP.

I. Padding

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

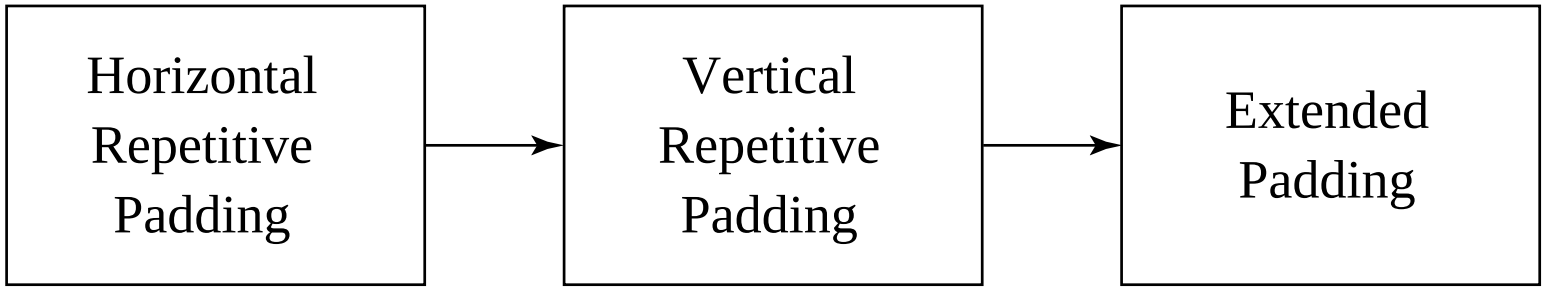


Fig. 12.6: A Sequence of Paddings for Reference VOPs in MPEG-4.

- 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 12.1 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 12.1: Repetitive Paddings

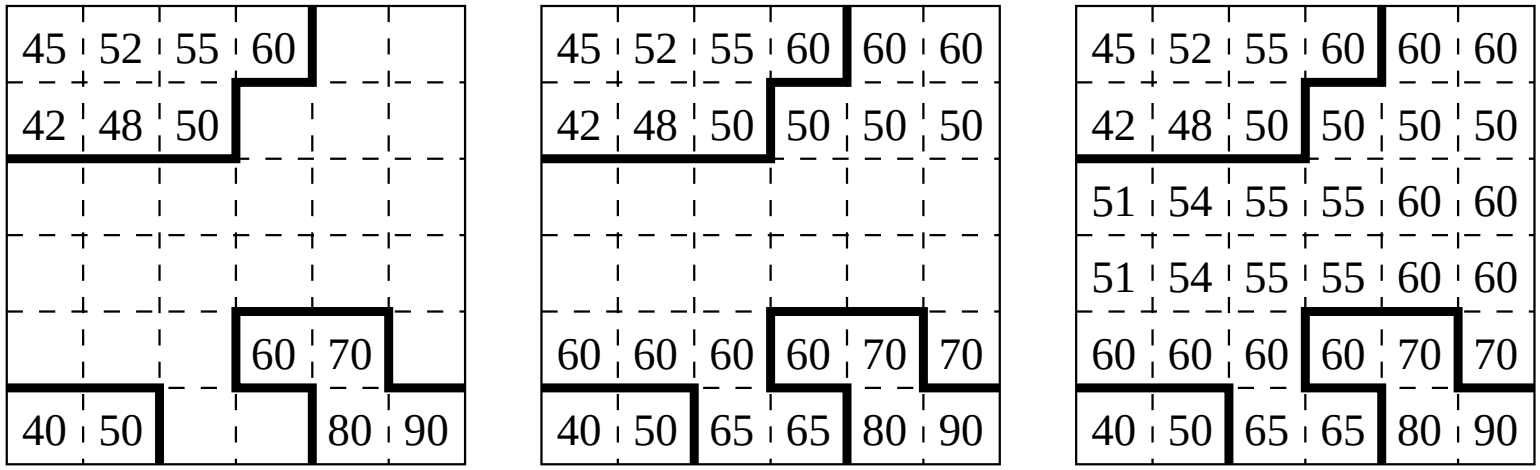


Fig. 12.7: An example of Repetitive Padding in a boundary macroblock of a Reference VOP: (a) Original pixels within the VOP, (b) After Horizontal Repetitive Padding, (c) Followed by Vertical Repetitive Padding.

II. 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:

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

  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 $\mathbf{MV}(u,v)$:

$$(u,v) = [\ (i,j) \mid SAD(i,j) \text{ is minimum, } i \in [-p,p],\ j \in [-p,p]\ ] \quad (12.1)$$

  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. 12.8 illustrates the process of texture coding for boundary MBs using the Shape Adaptive DCT (SA-DCT).

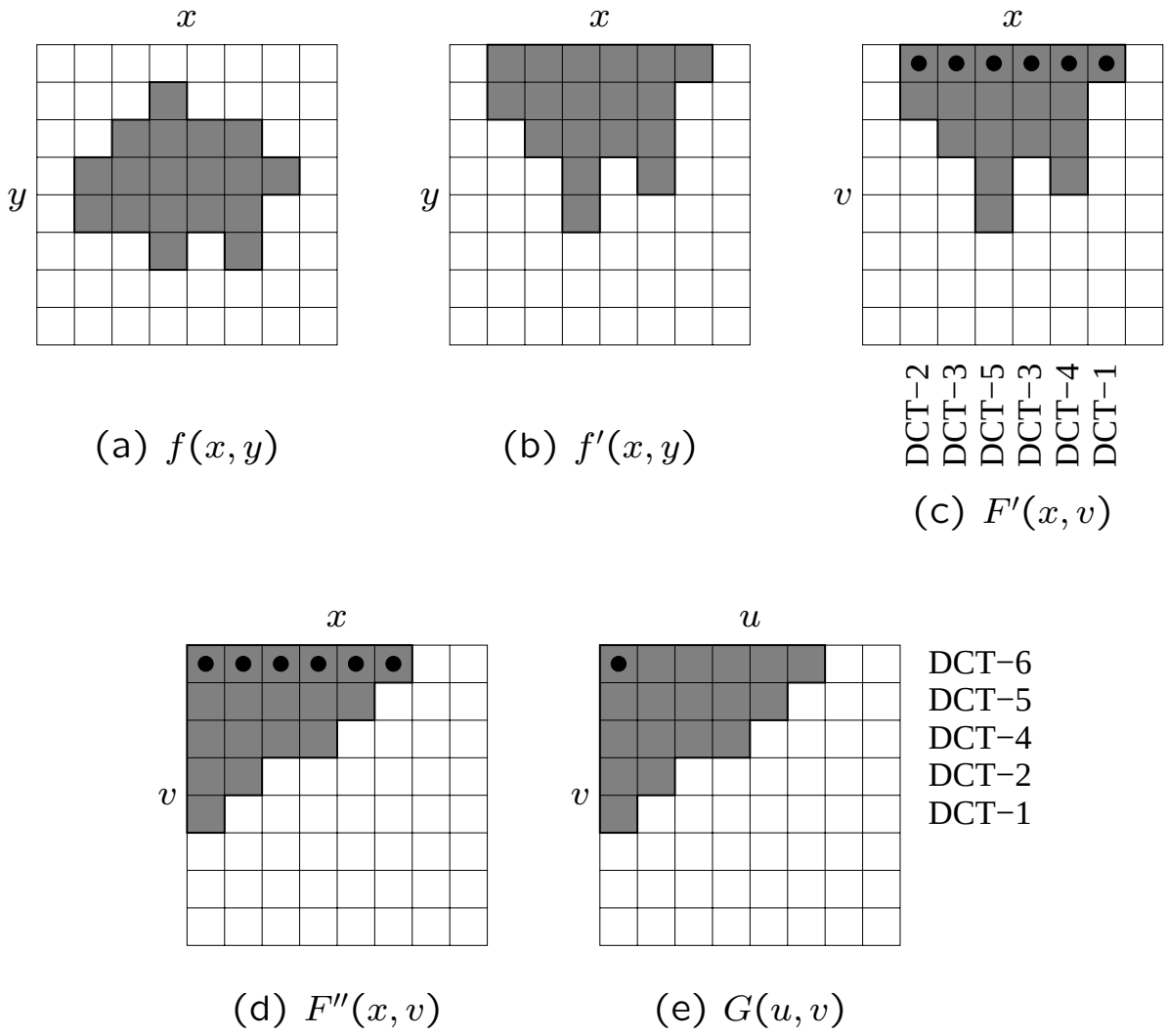


Fig. 12.8: Texture Coding for Boundary MBs Using the Shape Adaptive DCT (SA-DCT).

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).

I. 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)

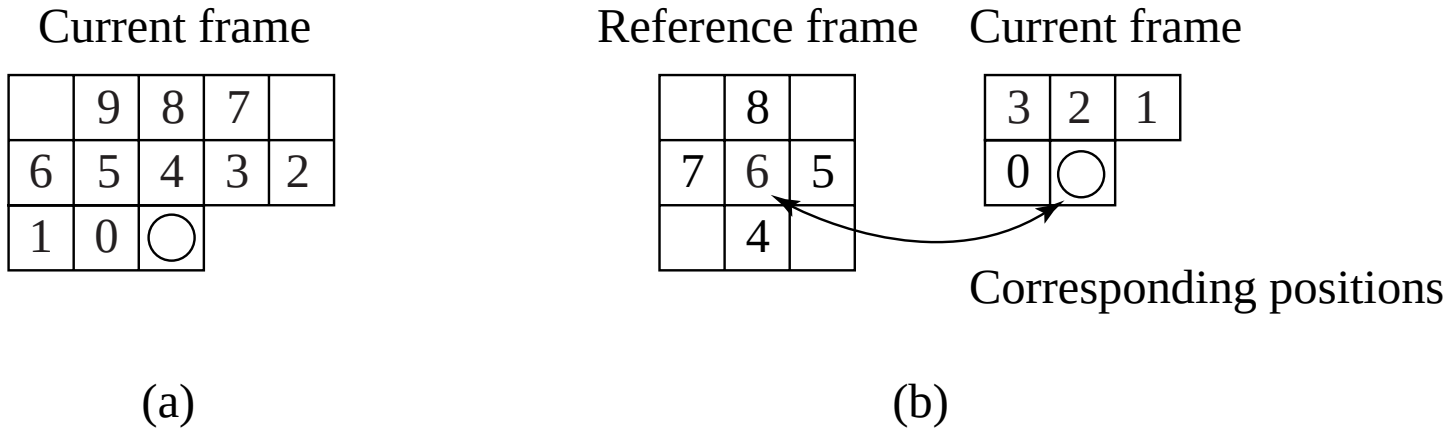


Fig. 12.9: Contexts in CAE for a pixel in the boundary BAB. (a) Intra-CAE, (b) Inter-CAE.

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.

II. 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 subband 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.

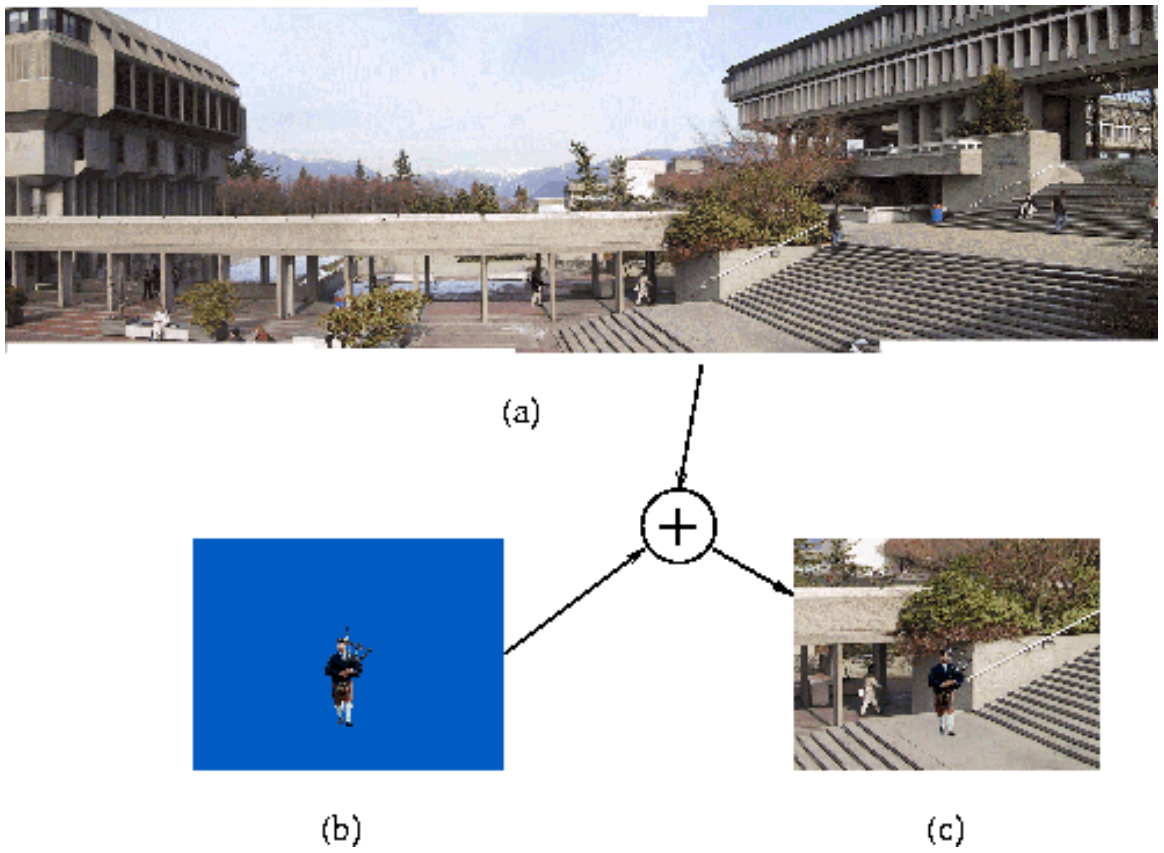


Fig. 12.10: Sprite Coding. (a) The sprite panoramic image of the background, (b) the foreground object (piper) in a blue-screen image, (c) the composed video scene.

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:

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

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

12.3 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)*.

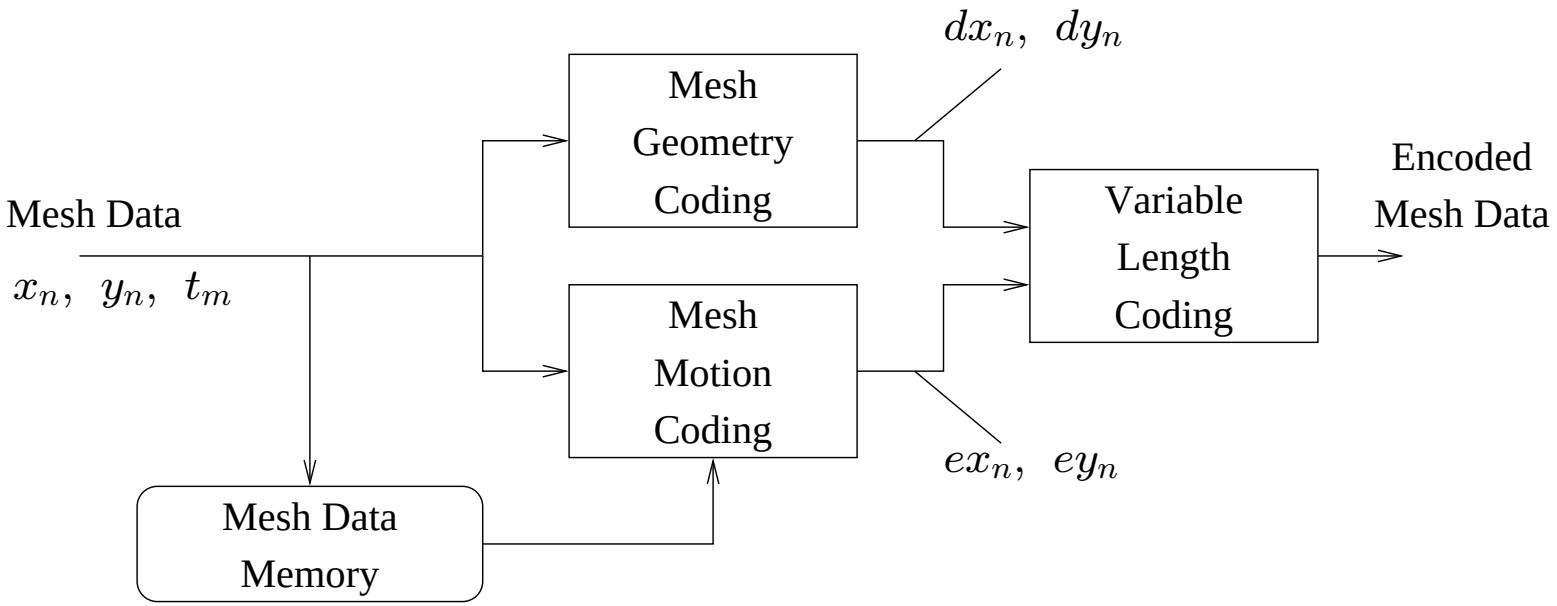
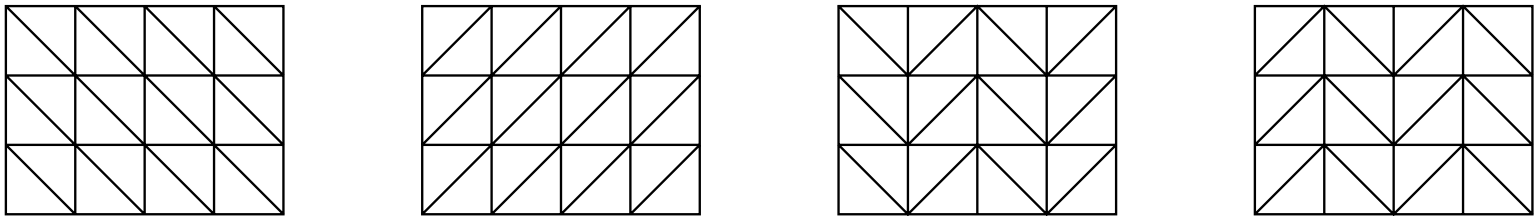


Fig. 12.11: 2D Mesh Object Plane (MOP) Encoding Process

I. 2D Mesh Geometry Coding

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



(a) Type 0 (b) Type 1 (c) Type 2 (d) Type 3

Fig. 12.12: Four Types of Uniform Meshes.

- **Definition:** If $\mathcal{D}$ is a Delaunay triangulation, then any of its triangles $t_n = (P_i,\ P_j,\ P_k) \in \mathcal{D}$ 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 mesh.
  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.

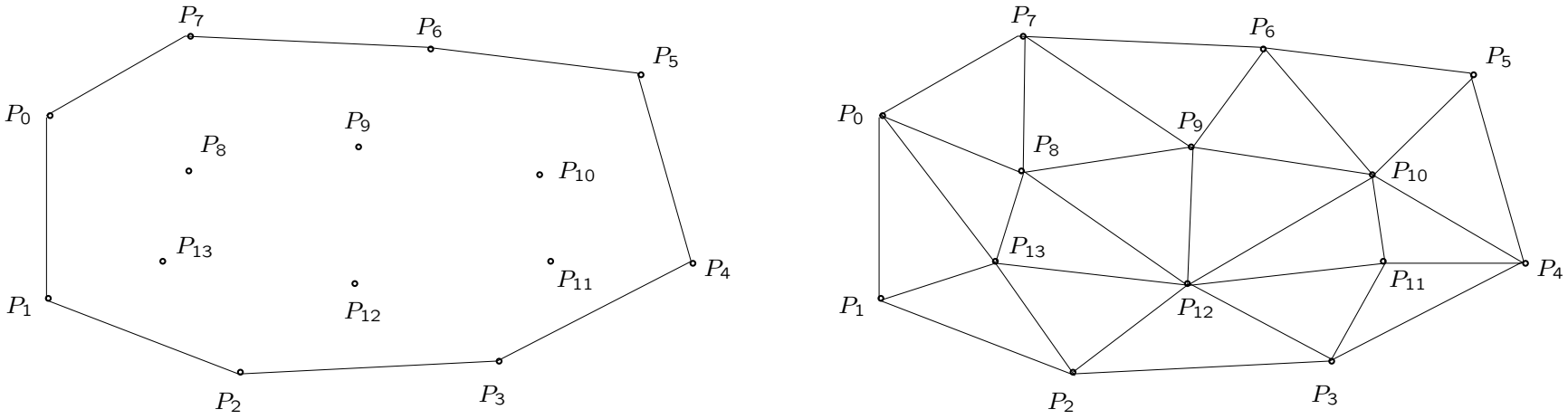


Fig. 12.13: Delaunay Mesh: (a) Boundary nodes (P_0 to P_7) and Interior nodes (P_8 to P_{13}). (b) Triangular mesh obtained by Constrained Delaunay Triangulation.

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

$$dx_n = x_n - x_{n-1}, \qquad dy_n = y_n - y_{n-1}, \tag{12.6}$$

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

II. 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 $\mathbf{MV_i}$ and $\mathbf{MV_j}$, then a prediction $\mathbf{Pred_k}$ will be made for the motion vector of P_k and this is rounded to a half-pixel precision:

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

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

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

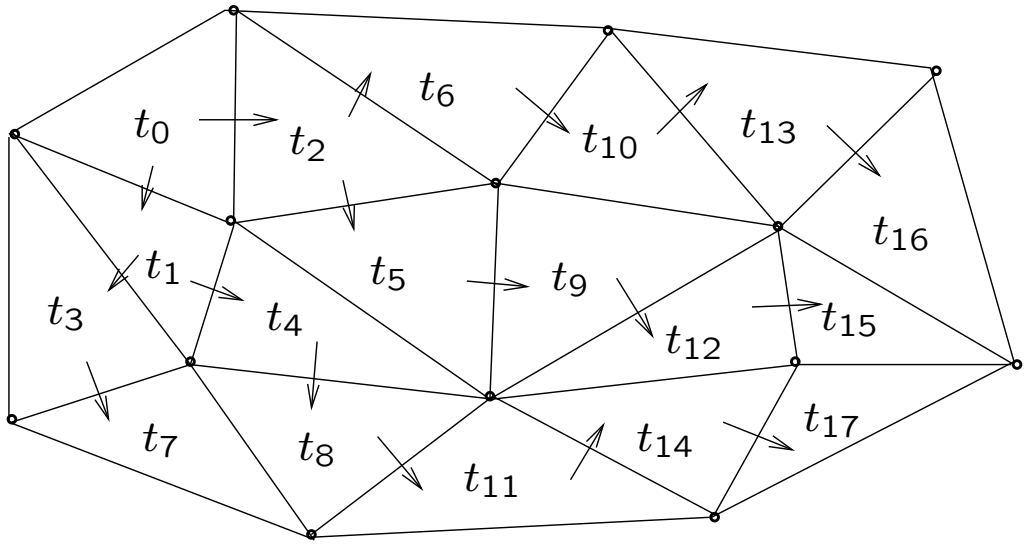
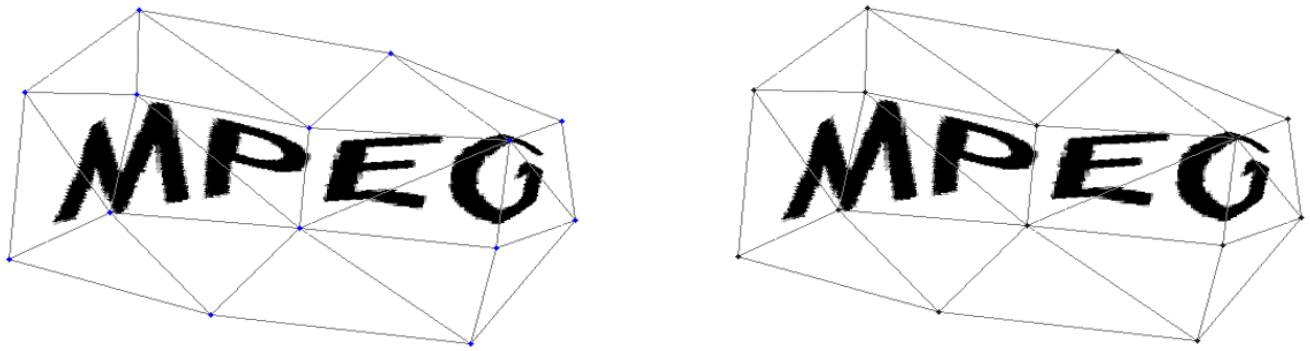


Fig. 12.14: A breadth-first order of MOP triangles for 2D mesh motion coding.



(a) (b)

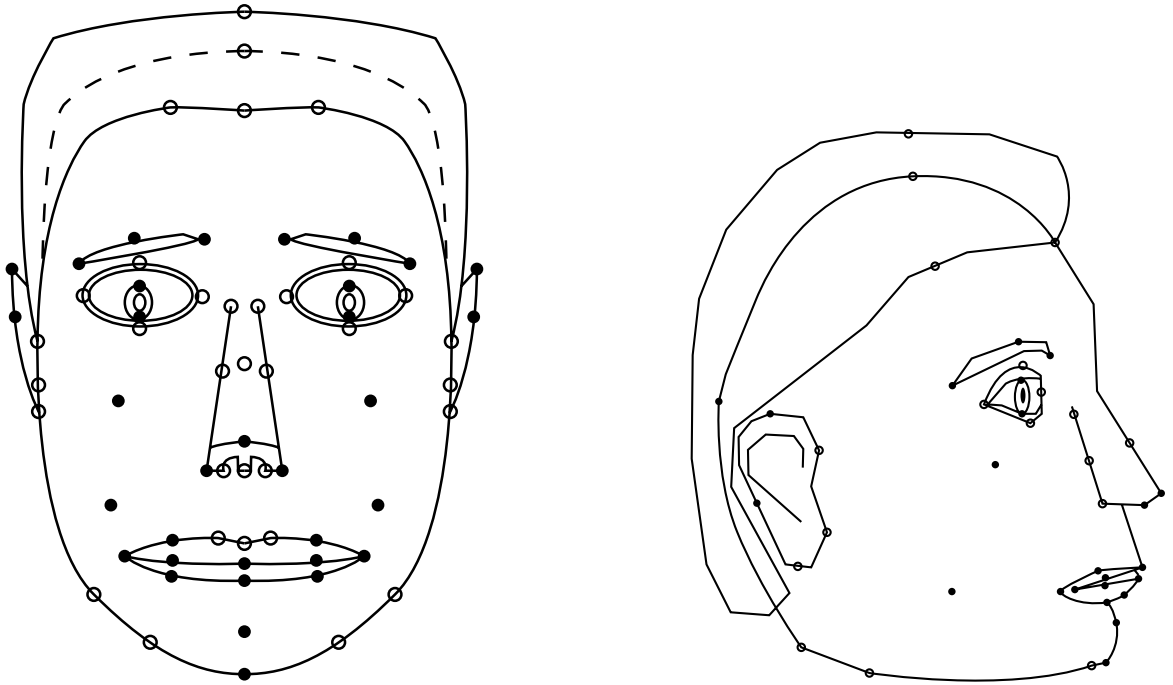
Fig. 12.15: Mesh-based texture mapping for 2D object animation.

12.3.2 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.

I. 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. 12.16 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.



(a) (b)

Fig. 12.16: Feature Points for Face Definition Parameters (FDPs). (Feature points for teeth and tongue not shown.)

II. 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.

12.4 MPEG-4 Object types, 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.

- **Object type** is introduced to define the tools needed to create video objects and how they can be combined in a scene.

Table 12.1: Tools for MPEG-4 Natural Visual Object Types

	Object Types					
				Simple		Scalable
Tools	Simple	Core	Main	scalable	N-bit	Still Texture
Basic MC-based tools	*	*	*	*	*	
B-VOP		*	*	*	*	
Binary Shape Coding		*	*		*	
Gray-level Shape Coding			*			
Sprite			*			
Interlace			*			
Temporal scalability (P-VOP)		*	*		*	
Spat. & Temp Scal. (r. VOP)				*		
N-bit					*	
Scalable Still Texture						*
Error Resilience	*	*	*	*	*	

Table 12.2: MPEG-4 Natural Visual Object Types and Profiles

Object Types	Profiles					
	Simple	Core	Main	Simple scalable	N-bit	Scalable Texture
Simple	*	*	*	*	*	
Core		*	*		*	
Main			*			
Simple Scalable				*		
N-bit					*	
Scalable Still Texture			*			*

- For "Main Profile", for example, only Object Types "Simple", "Core", "Main", and "Scalable Still Texture" are supported.

Table 12.3: MPEG-4 Levels in Simple, Core, and Main Visual Profiles

Profile	Level	Typical picture size	Bit-rate (bits/sec)	Max number of objects
Simple	1	176×144 (QCIF)	64 k	4
	2	352×288 (CIF)	128 k	4
	3	352×288 (CIF)	384 k	4
Core	1	176×144 (QCIF)	384 k	4
	2	352×288 (CIF)	2 M	16
Main	1	352×288 (CIF)	2 M	16
	2	720×576 (CCIR601)	15 M	32
	3	1920×1080 (HDTV)	38.4 M	32

12.5 MPEG-4 Part10/H.264

- The H.264 video compression standard, formerly known as "H.26L", is being developed by the Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG.
- Preliminary studies using software based on this new standard suggests that H.264 offers up to 30-50% better compression than MPEG-2, and up to 30% over H.263+ and MPEG-4 advanced simple profile.
- The outcome of this work is actually two identical standards: **ISO MPEG-4 Part10** and **ITU-T H.264**.
- H.264 is currently one of the leading candidates to carry High Definition TV (HDTV) video content on many potential applications.

- **Core Features**

  - VLC-Based Entropy Decoding:

    Two entropy methods are used in the variable-length entropy decoder: Unified-VLC (UVLC) and Context Adaptive VLC (CAVLC).

  - Motion Compensation (P-Prediction):

    Uses a tree-structured motion segmentation down to 4×4 block size (16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4). This allows much more accurate motion compensation of moving objects. Furthermore, motion vectors can be up to half-pixel or quarter-pixel accuracy.

  - Intra-Prediction (I-Prediction):

    H.264 exploits much more spatial prediction than in previous video standards such as H.263+.

- Uses a simple integer-precision 4×4 DCT, and a quantization scheme with nonlinear step-sizes.
- In-Loop Deblocking Filters.

- **Baseline Profile Features**

  The Baseline profile of H.264 is intended for real-time conversational applications, such as videoconferencing.

  It contains all the core coding tools of H.264 discussed above and the following additional error-resilience tools, to allow for error-prone carriers such as IP and wireless networks:

  - Arbitrary slice order (ASO).
  - Flexible macroblock order (FMO).
  - Redundant slices.

- **Main Profile Features**

  Represents non-low-delay applications such as broadcasting and stored-medium.

  The Main profile contains all Baseline profile features (except ASO, FMO, and redundant slices) plus the following:

  – B slices.

  – Context Adaptive Binary Arithmetic Coding (CABAC).

  – Weighted Prediction.

- **Extended Profile Features**

  The eXtended profile (or profile X) is designed for the new video streaming applications. This profile allows non-low-delay features, bitstream switching features, and also more error-resilience tools.

12.6 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. 12.17 illustrates some possible applications that will benefit from the MPEG-7 standard.

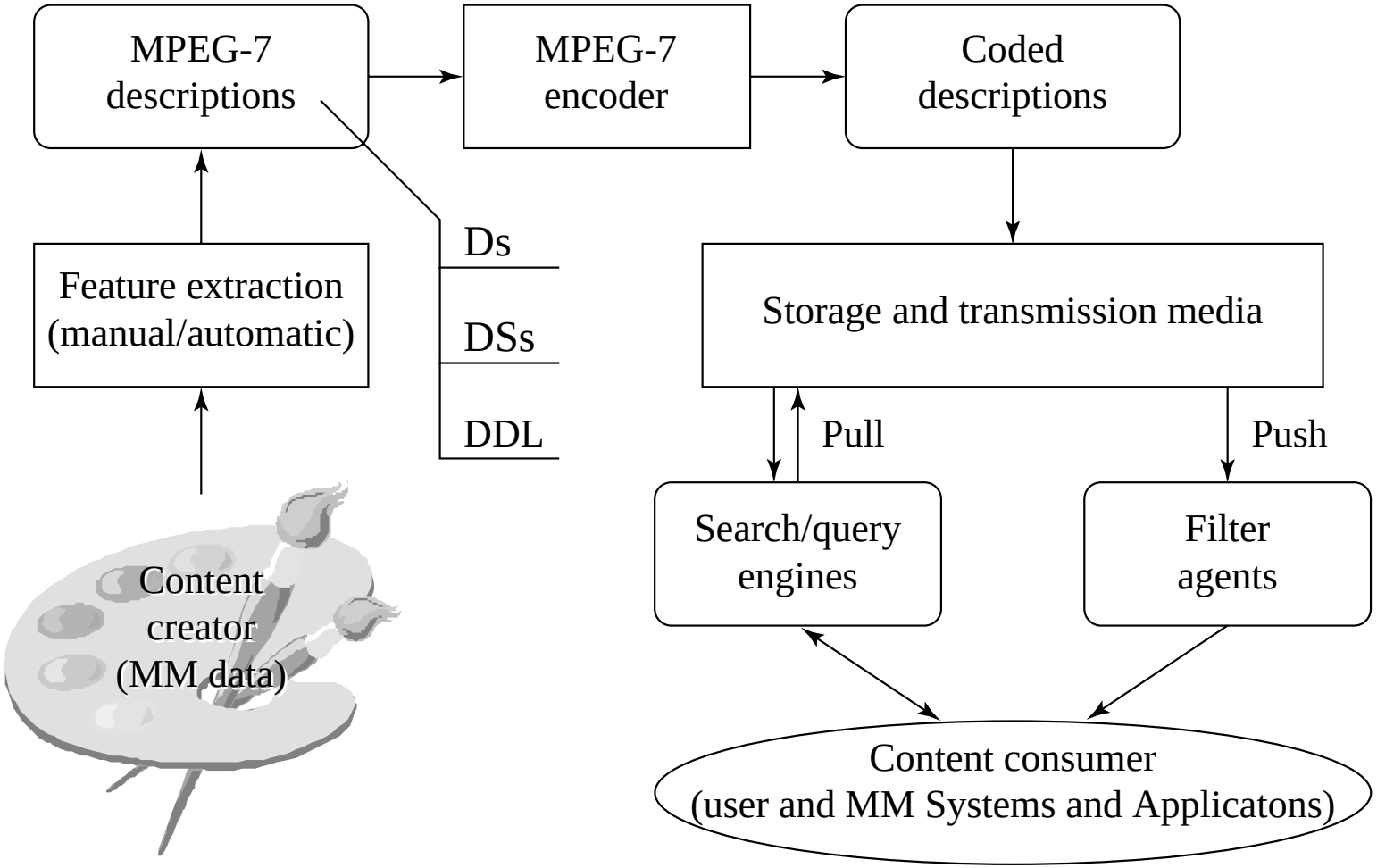


Fig. 12.17: Possible Applications using MPEG-7.

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. 12.18).
  - Object motion trajectory.
  - Parametric object motion.
  - Motion activity.
- **Localization**
  - Region locator.
  - Spatiotemporal locator.
- **Others**
  - Face recognition.

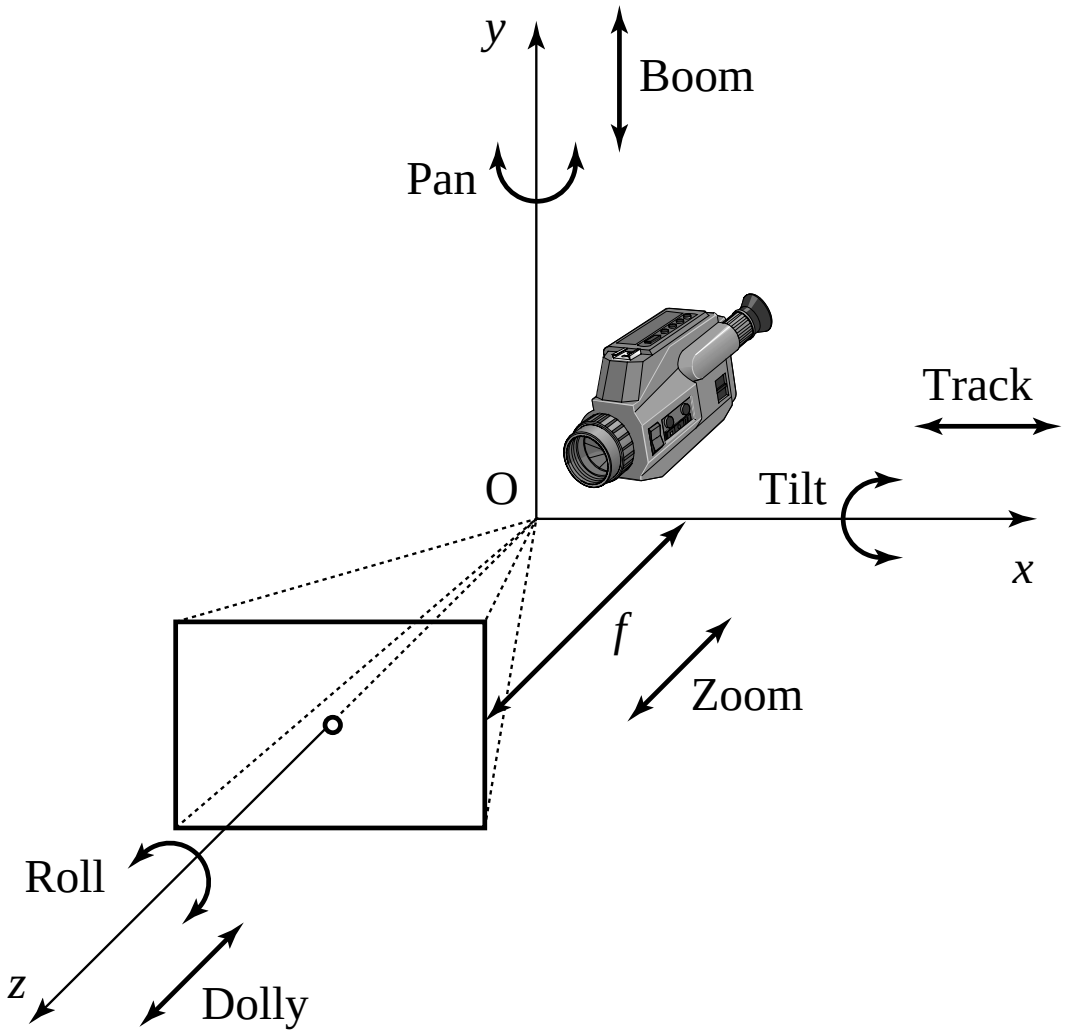


Fig. 12.18: Camera motions: pan, tilt, roll, dolly, track, and boom.

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**
  - UserPreference.

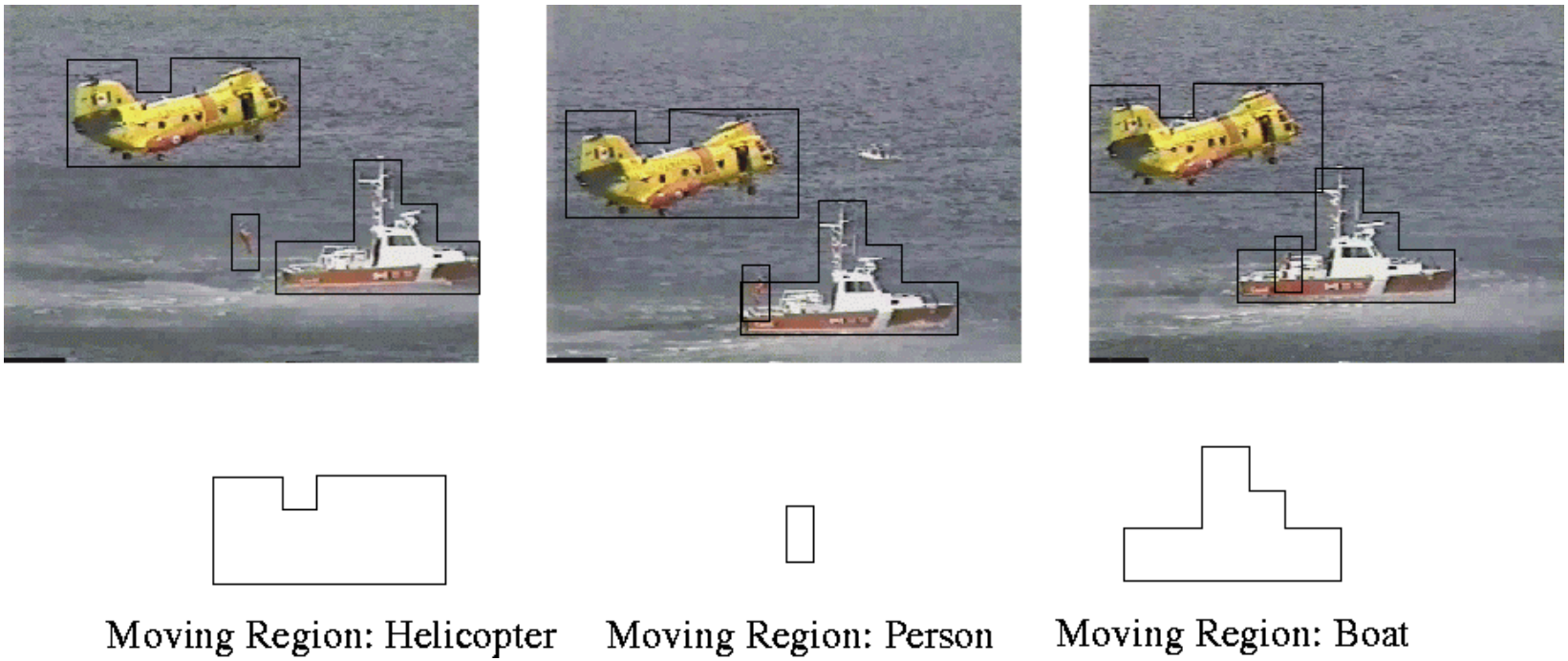


Fig. 12.19: MPEG-7 video segment.

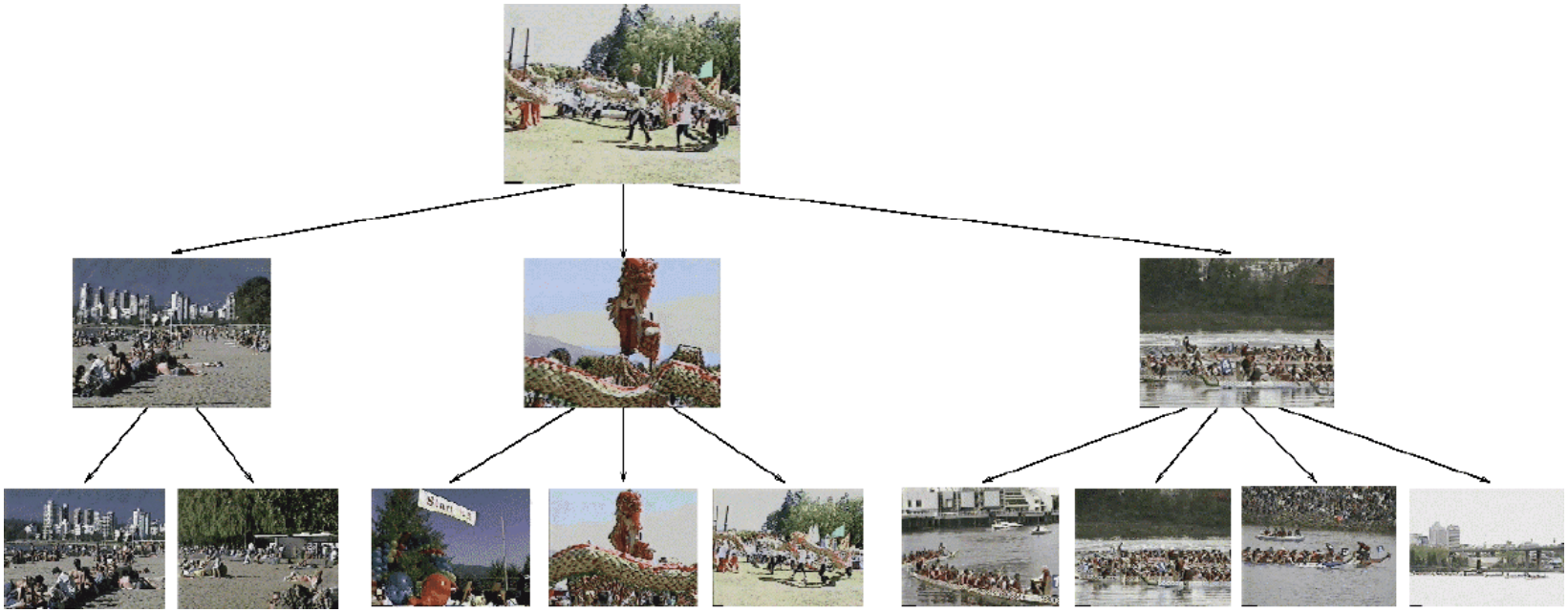


Fig. 12.20: A video summary.

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.

12.7 MPEG-21

- The development of the newest standard, **MPEG-21: Multimedia Framework**, started in June 2000, and was expected to become International Stardard by 2003.
- The *vision* for MPEG-21 is to define a multimedia framework to enable transparent and augmented use of multimedia resources across a wide range of networks and devices used by different communities.
- The seven key elements in MPEG-21 are:
  - **Digital item declaration** — to establish a uniform and flexible abstraction and interoperable schema for declaring Digital items.
  - **Digital item identification and description** — to establish a framework for standardized identification and description of digital items regardless of their origin, type or granularity.

- **Content management and usage** — to provide an interface and protocol that facilitate the management and usage (searching, caching, archiving, distributing, etc.) of the content.
- **Intellectual property management and protection (IPMP)** — to enable contents to be reliably managed and protected.
- **Terminals and networks** — to provide interoperable and transparent access to content with Quality of Service (QoS) across a wide range of networks and terminals.
- **Content representation** — to represent content in an adequate way for pursuing the objective of MPEG-21, namely "content anytime anywhere".
- **Event reporting** — to establish metrics and interfaces for reporting *events* (user interactions) so as to understand performance and alternatives.

12.8 Further Exploration

- **Text books:**
  - *Multimedia Systems, Standards, and Networks* by A. Puri and T. Chen
  - *The MPEG-4 Book* by F. Pereira and T. Ebrahimi
  - *Introduction to MPEG-7: Multimedia Content Description Interface* by B.S. Manjunath et al.
- **Web sites:** ⟶ Link to Further Exploration for Chapter 12.. including:
  - The MPEG home page
  - The MPEG FAQ page
  - Overviews, tutorials, and working documents of MPEG-4
  - Tutorials on MPEG-4 Part 10/H.264
  - Overviews of MPEG-7 and working documents for MPEG-21
  - Documentation for XML schemas that form the basis of MPEG-7 DDL