Video Coding

Video Compression

John G. Apostolopoulos
Streaming Media Systems Group
Hewlett-Packard Laboratories
japos@hpl.hp.com

Outline of Today's Lecture

- Motivation for video compression
- Brief review of image compression (two previous lectures)
- Video compression
  - Motion-compensated prediction
  - Motion-compensated interpolation
  - Generic (MPEG-type) video coder architecture
  - Scalable video coding
- Current video compression standards
  - What do the standards specify?
  - Frame-based video coding: MPEG-1/2/4, H.261/3
  - Object-based video coding: MPEG-4

Video Coding

Motivation for Video Compression

- Problem:
  - Raw video contains an immense amount of data
  - Communication and storage capabilities are limited and expensive
- Example HDTV video signal:
  - 720x1280 pixels/frame, progressive scanning at 60 frames/s:

$$\left(\frac{720\times1280\,pixels}{frame}\right)\left(\frac{60\,frames}{\text{sec}}\right)\left(\frac{3colors}{pixel}\right)\left(\frac{8bits}{color}\right)=1.3Gb/s$$

  - 20 Mb/s HDTV channel bandwidth

  $\rightarrow$ Requires compression by a factor of 70 (equivalent to .35 bits/pixel)

Video Coding

Achieving Compression

- Reduce *redundancy* and *irrelevancy*
- *Sources of redundancy*
  - Temporal: Adjacent frames highly correlated
  - Spatial: Nearby pixels are often correlated with each other
  - Color space: RGB components are correlated among themselves

  → Relatively straightforward to exploit
- *Irrelevancy*
  - Perceptually unimportant information

  → Difficult to model and exploit

Video Coding

Spatial and Temporal Redundancy

- Why can video be compressed?
  - Video contains much spatial and temporal redundancy.

- *Spatial redundancy:* Neighboring pixels are similar
- *Temporal redundancy:* Adjacent frames are similar

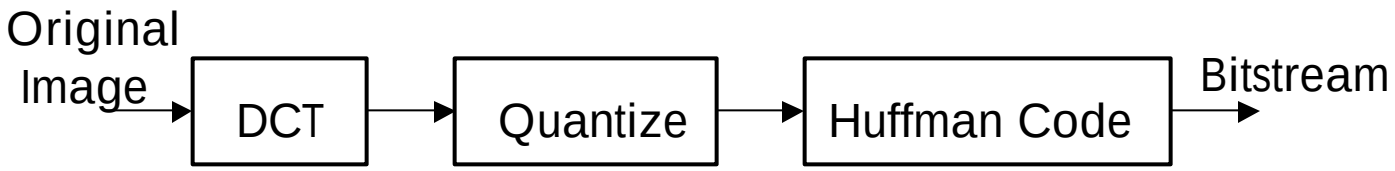
Compression is achieved by exploiting the spatial and temporal redundancy inherent to video.

Video Coding

Image Compression (Review of Last Lecture)

- Coding a single frame (image):
  - Partition image into 8x8-pixel blocks
  - Compute 2-D Discrete Cosine Transform (DCT) of each block
  - Quantize each DCT coefficient
  - Runlength and Huffman code the nonzero quantized DCT coefficients

→Basis for the JPEG Image Compression Standard



- Color Image:
  - Begin with RGB to luminance/chrominance conversion

Video Coding

Video Compression

- *Video*: *Sequence of frames* (images) that are *related*
- Related along the temporal dimension
  - Therefore temporal redundancy exists
- Main addition over image compression
  - Temporal redundancy

  → *Exploit the temporal redundancy*

Video Coding

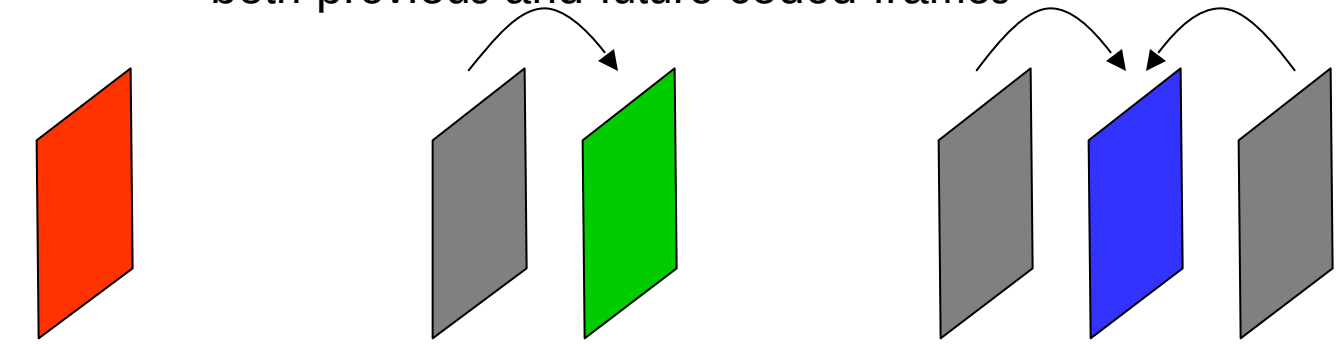
Temporal Processing

- Usually high frame rate: *Significant temporal redundancy*
- Possible representations along temporal dimension:
  - *Transform/subband methods*
    - Good for ideal case (constant velocity uniform global motion)
    - Inefficient for nonuniform motion
    - Requires large number of frame stores
      - Leads to delay (Memory cost may also be an issue)
  - *Predictive methods*
    - Good performance using only 2 frame stores
    - However, simple frame differencing in not enough

Video Coding

Video Compression

- Main addition over image compression:
  - Exploit the temporal redundancy
- *Predict current frame* based on previously coded frames
- Types of coded frames:
  - *I-frame:* Intra-coded frame, coded independently of all other frames
  - *P-frame:* Predictively coded frame, coded based on previously coded frame
  - *B-frame:* Bidirectionally predicted frame, coded based on both previous and future coded frames



I frame **P-frame** **B-frame**

Video Coding

Temporal Processing: Motion-Compensated Prediction

- Simple frame differencing *fails* when there is motion
- *Must account for motion*
  - → *Motion-compensated (MC) prediction*
- MC-prediction generally provides significant improvements
- Question:
  - How can we estimate motion?
  - How can we form MC-prediction?

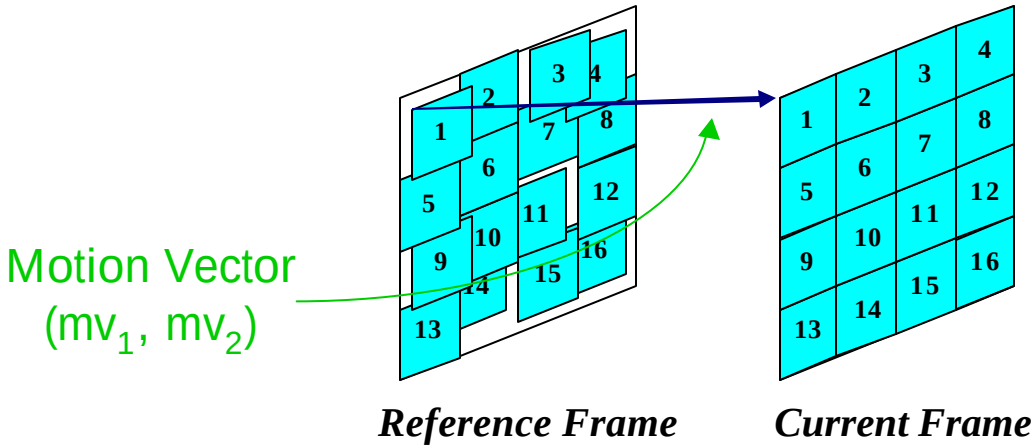
Video Coding

Temporal Processing: Motion Estimation

- Ideal situation:
  - Partition video into moving objects
  - Describe object motion
  - → Very difficult
- Practical approach: *Block-matching ME*
  - Partition each frame into blocks
  - Describe motion of each block
  - → No object identification required
  - → Good, robust performance

Video Coding

Block Matching Algorithm



- *Assumptions:*
  - Translational motion over local region:

$$f(n_1, n_2, k_{cur}) = f(n_1 - mv_1, n_2 - mv_2, k_{ref})$$

  - Pixels within a block have the same motion
- *ME Algorithm:*
  - Divide current frame into non-overlapping $N_1 x N_2$ blocks
  - For each block, find the *best matching block* in reference frame
- *MC-Prediction Algorithm:*
  - Use best matching blocks of reference frame as prediction of blocks in current frame

Motion Vectors and Motion Vector Field

- *Motion vector*
  - Expresses the *relative horizontal and vertical offsets* *(mv_1,mv_2)*, or motion, of a given block from one frame to another
  - Each block has its own motion vector
- *Motion vector field*
  - Collection of motion vectors for all the blocks in a frame

Video Coding

Block Matching: Determining the Best Matching Block

- For each block in the current frame search for best matching block in the reference frame
  - *Metrics* for determining "best match":

$$MSE = \sum_{(n_1,n_2)\in Block} \sum \left[f(n_1,n_2,k_{cur}) - f(n_1 - mv_1, n_2 - mv_2, k_{ref}) \right]^2$$

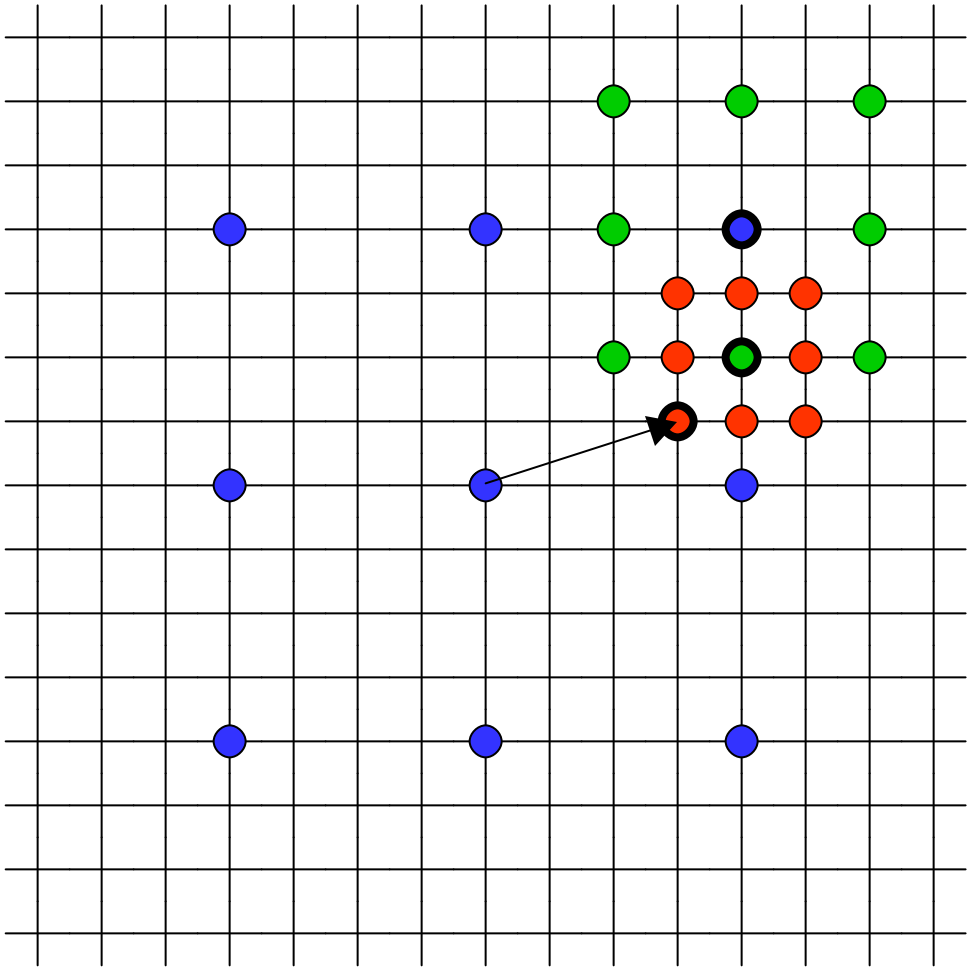
$$MAE = \sum_{(n_1,n_2)\in Block} \sum \left| f(n_1,n_2,k_{cur}) - f(n_1 - mv_1, n_2 - mv_2, k_{ref}) \right|$$

  - *Candidate matches*: $\text{All vectors in, e.g., } (\pm 32, \pm 32) \text{ pixel area}$
  - *Strategies for searching* candidate vectors for best match
    - Full search: Examine all candidate vectors
    - Partial (fast) search: Examine a carefully selected subset
- Best estimate of motion for each block: *"motion vector"*

[Section 8.4.2 of class textbook]

Video Coding

Example of Fast Search: 3-Step (Log) Search



- *Goal: Reduce number of search points*
- Example: $(\pm 7, \pm 7)$ search area
- Dots represent search points
- Search performed in 3 steps (coarse-to-fine):
  - Step 1: ● $(\pm 4\, pixels)$
  - Step 2: ● $(\pm 2\, pixels)$
  - Step 3: ● $(\pm 1\, pixels)$
- Best match is found at each step
- Next step: Search is centered around the best match of prior step
- Speedup increases for larger search areas

Video Coding

Half-Pixel Motion Estimation

- Motivation:
  - *Motion is not limited to integer-pixel offsets*
  - However, video only known at discrete pixel locations
  - To estimate sub-pixel motion, frames must be *spatially interpolated*
- Fractional MVs are used to represent the sub-pixel motion.
- Improved performance (extra complexity is worthwhile)
- Used in most standards: MPEG-1 / 2 / 4
- *Why are half-pixel motion vectors better?*
  - Can capture half-pixel motion
  - Averaging effect (from spatial interpolation) reduces prediction error → Improved prediction
  - For noisy sequences, averaging effect reduces noise → Improved compression

Video Coding

Practical Half-Pixel Motion Estimation Algorithm

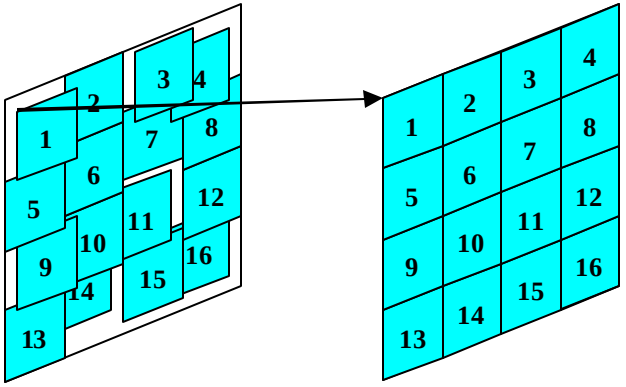
- *Half-pixel ME (coarse-fine) algorithm:*
  - *Coarse:* Perform integer motion estimation on blocks; find best integer-pixel MV
  - *Fine:* Refine estimate to find best half-pixel MV
    - Spatially interpolate the selected block in reference frame
    - Compare current block to interpolated reference frame block
    - Choose the integer or half-pixel offset that provides best match
- Typically, bilinear interpolation is used for spatial interpolation

Video Coding

Example: MC-Prediction for Two Consecutive Frames

Previous Frame (Reference Frame)

Current Frame (To be Predicted)

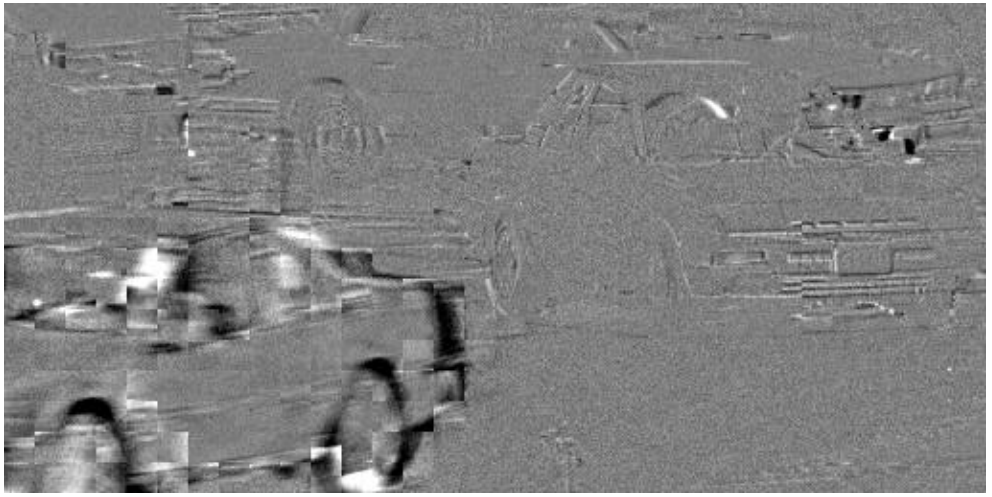


Reference Frame ***Predicted Frame***

Example: MC-Prediction for Two Consecutive Frames (cont.)

Prediction of Current Frame

Prediction Error (Residual)

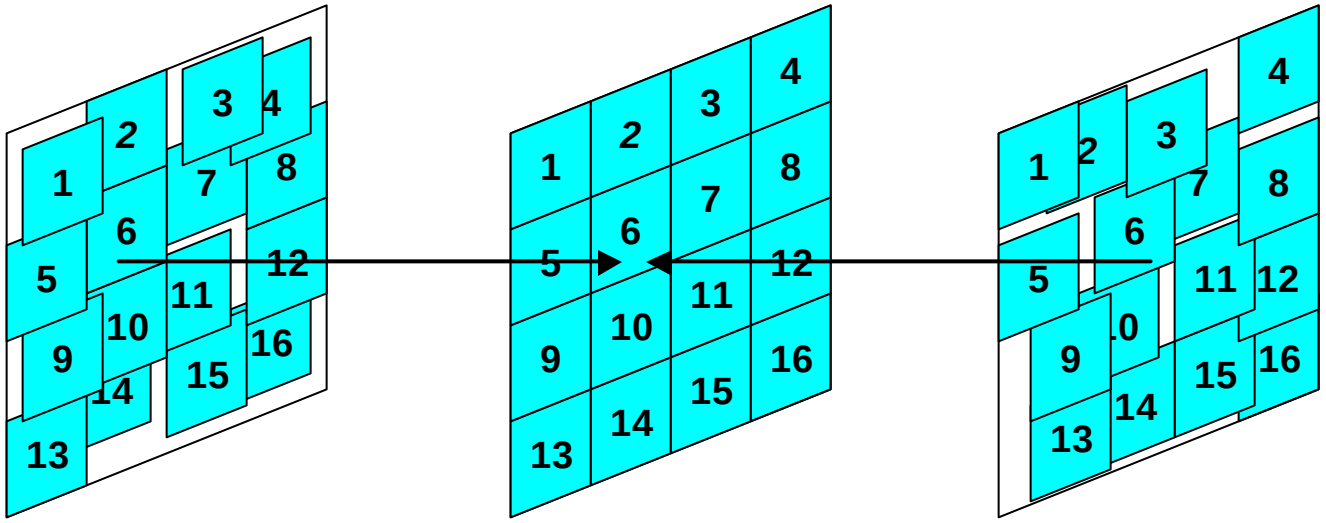


Block Matching Algorithm (cont.)

- *Issues:*
  - Block size?
  - Search range?
  - Motion vector accuracy?
- *Advantages:*
  - Good, robust performance for compression
  - Resulting motion vector field is easy to represent (one MV per block) and useful for compression
  - Simple, periodic structure, easy VLSI implementations
- *Disadvantages:*
  - Assumes translational motion model → Breaks down for more complex motion
  - Often produces blocking artifacts (OK for coding with Block DCT)

Video Coding

Motion-Compensated Interpolation (Bi-directional Prediction)

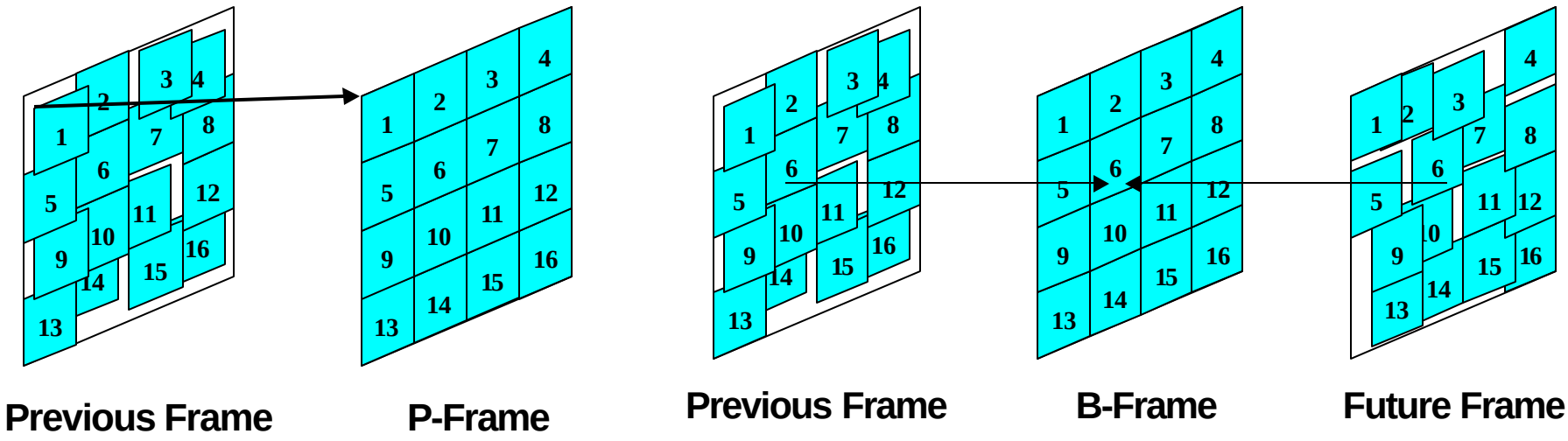


- *MC-interpolation* (or *bi-directional prediction*) is used to estimate a block in the current frame from a block in:
  1) Previous frame
  2) Future frame
  3) Average of a block from the previous frame and a block from the future frame

Video Coding

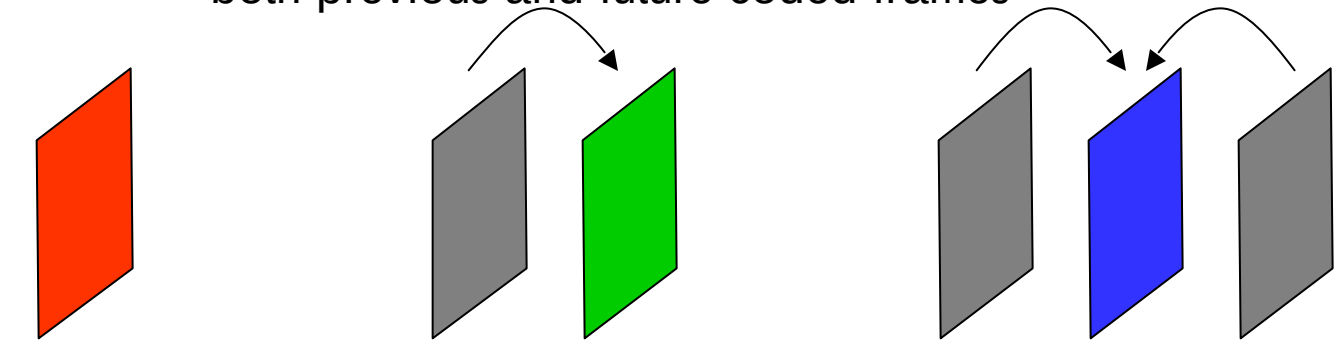
MC-Prediction and Bi-directional Prediction (P- and B-frames)

- Motion compensated prediction: Predict the current frame based on reference frame(s) while compensating for the motion
- Examples of block-based motion-compensated prediction (*P-frame*) and bi-directional prediction (*B-frame*):



Video Compression

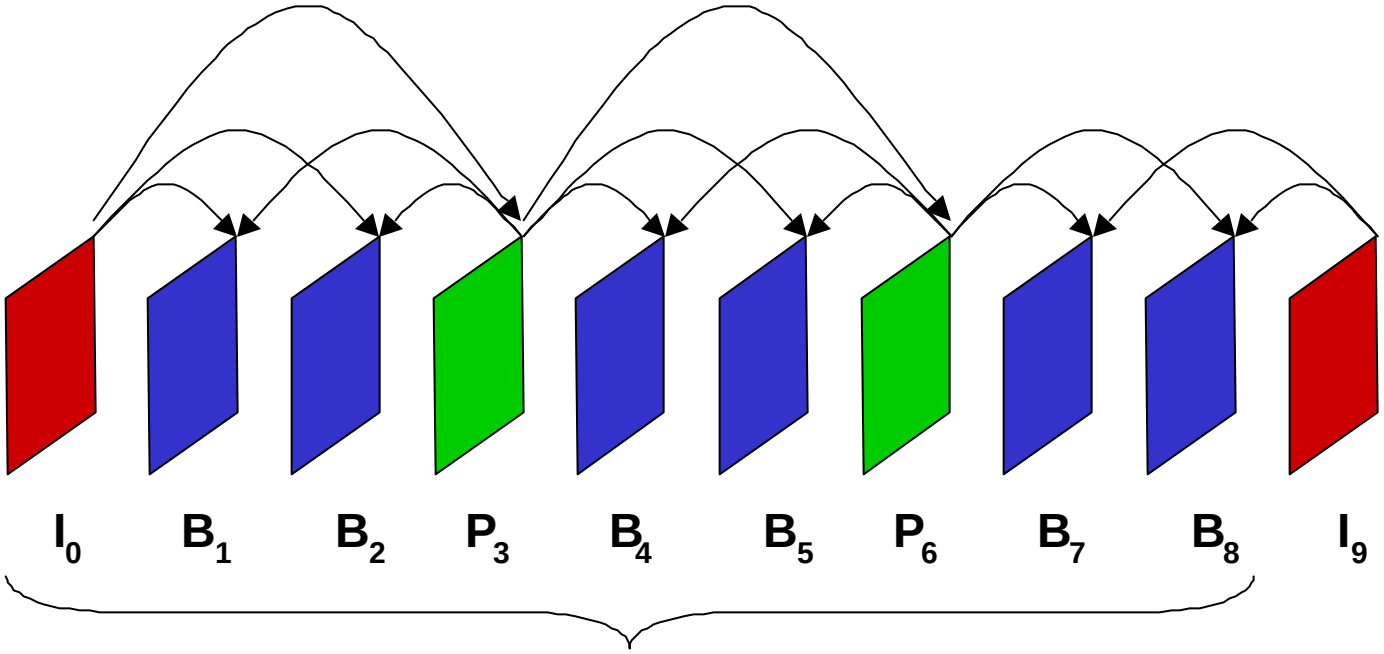
- Main addition over image compression:
  - Exploit the temporal redundancy
- Predict current frame based on previously coded frames
- Types of coded frames:
  - *I-frame:* Intra-coded frame, coded independently of all other frames
  - *P-frame:* Predictively coded frame, coded based on previously coded frame
  - *B-frame:* Bidirectionally predicted frame, coded based on both previous and future coded frames



I frame **P-frame** **B-frame**

Example Use of I-,P-,B-frames: MPEG Group of Pictures (GOP)

- Arrows show prediction dependencies between frames



Video Coding

Summary of Temporal Processing

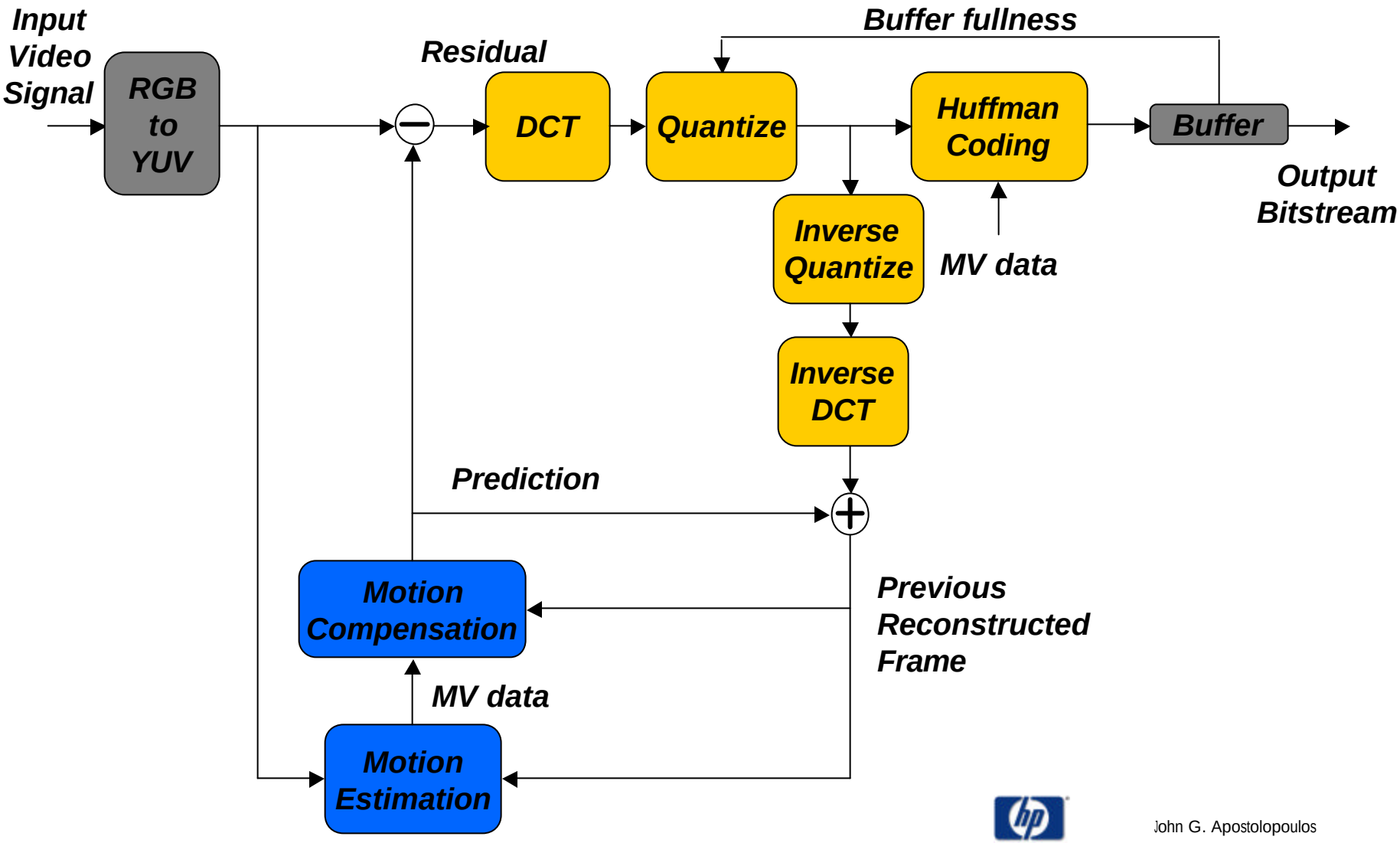
- Use MC-prediction and MC-interpolation to reduce temporal redundancy
- MC-P and MC-I usually perform well; In compression have a second chance to recover when they perform badly
- *MC-P and MC-I yields:*
  - *Motion vectors*
  - *MC-P error or residual → Code error with conventional image coder*
- Sometimes MC-P and MC-I may *perform badly*
  - Examples: Complex motion, new imagery (occlusions)
  - Approach:
    1. Identify blocks where prediction fails
    2. Code block *without prediction*

Video Coding

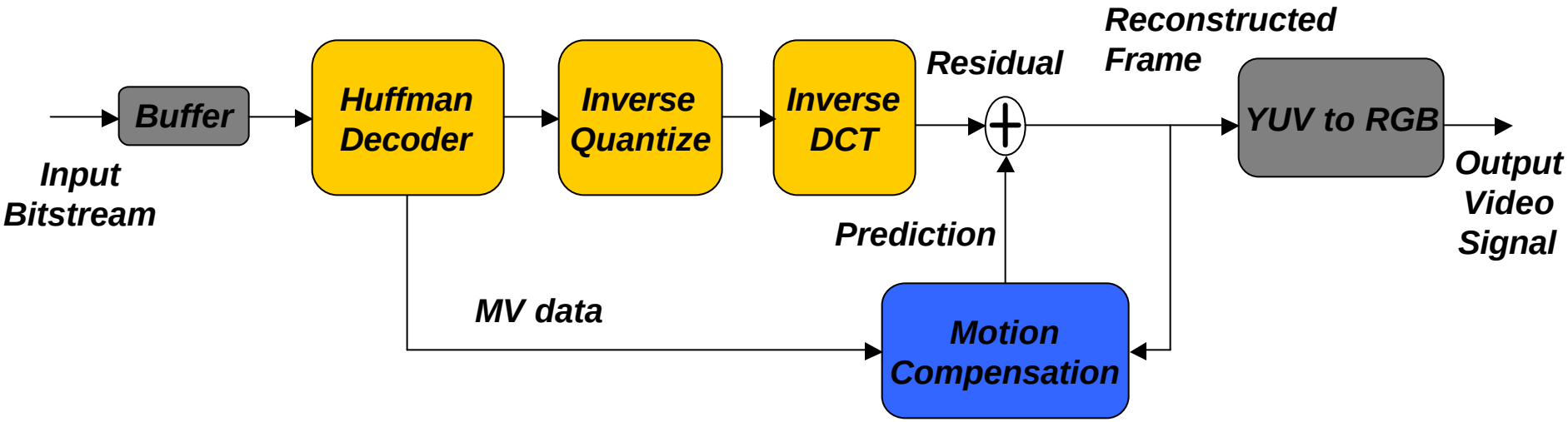
Basic Video Compression Architecture

- Exploiting the redundancies:
  - Temporal: MC-prediction and MC-interpolation
  - Spatial: Block DCT
  - Color: Color space conversion
- Scalar quantization of DCT coefficients
- Zigzag scanning, runlength and Huffman coding of the nonzero quantized DCT coefficients

Example Video Encoder



Example Video Decoder



Video Coding

Scalable Video Coding

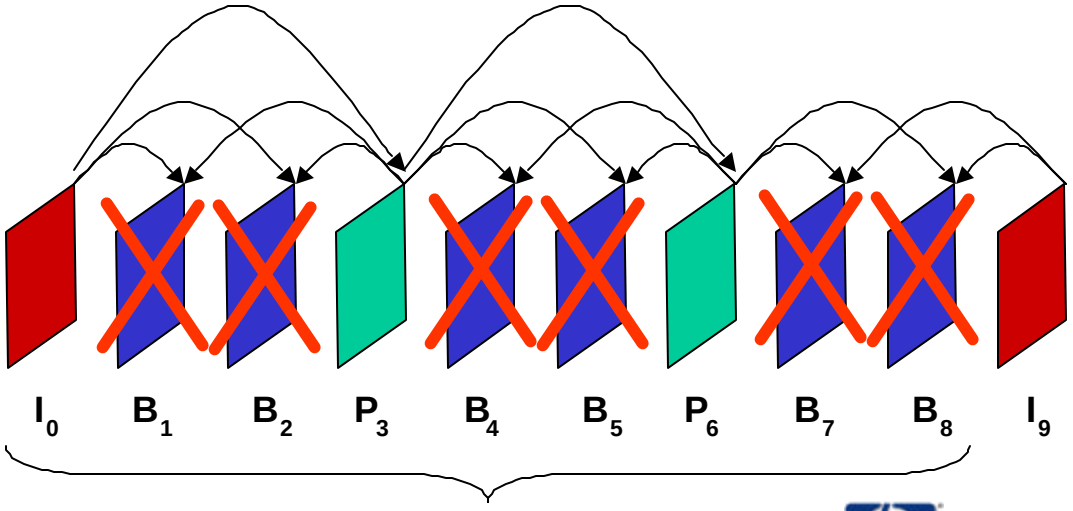
- *Scalable coding:*
  - Decompose video into *layers of prioritized importance*
  - Code layers into *base and enhancement* bitstreams
  - Use *one or more bitstreams* to reconstruct video of increasing quality
- Scalability with respect to: Number of layers that may be decoded, required bandwidth, required computation/memory
- *Types of Scalability:*
  - Temporal scalability
  - Spatial scalability
  - PSNR (quality) scalability

Scalable Video Coding (cont.)

- Types of Scalability:
  - *Temporal scalability* → *Temporal resolution*
  - *Spatial scalability* → *Spatial resolution*
  - *PSNR (quality) scalability* → *Amplitude resolution*
- Each form of scalable video coding provides scalability of one dimension of the video signal

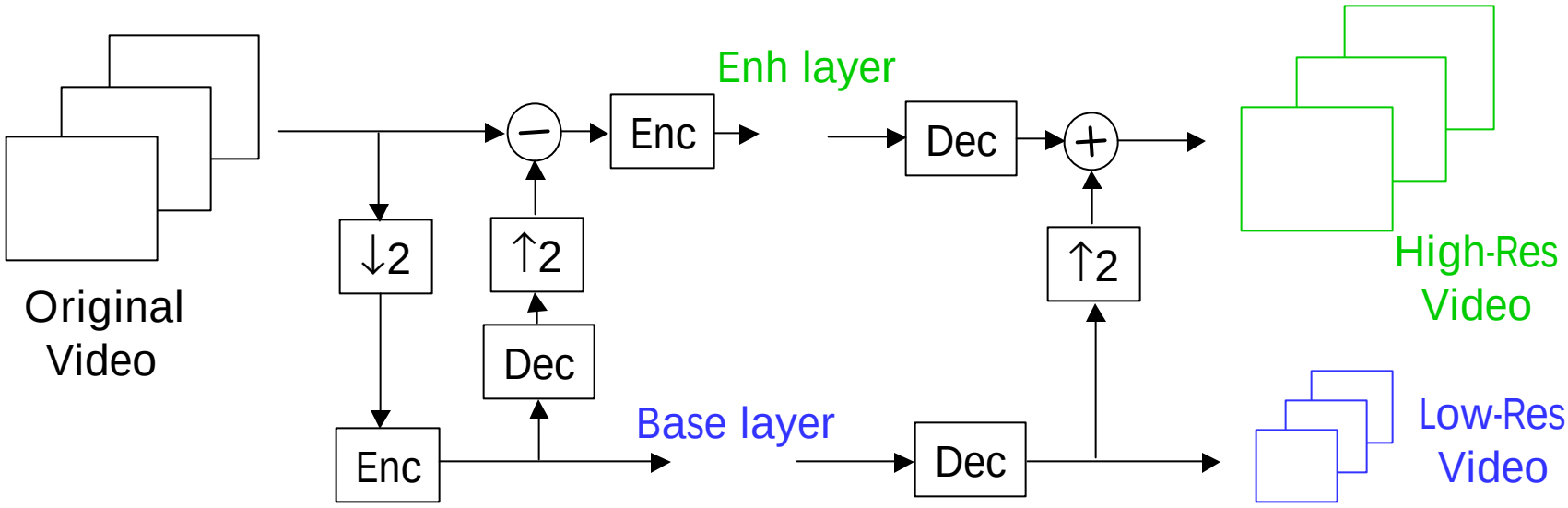
Scalable Coding: Temporal Scalability

- *Temporal scalability:* Based on the use of *B-frames* to refine the *temporal resolution*
  - B-frames are dependent on other frames
  - However, *no other frame depends on a B-frame*
  - Each *B-frame may be discarded* without affecting other frames



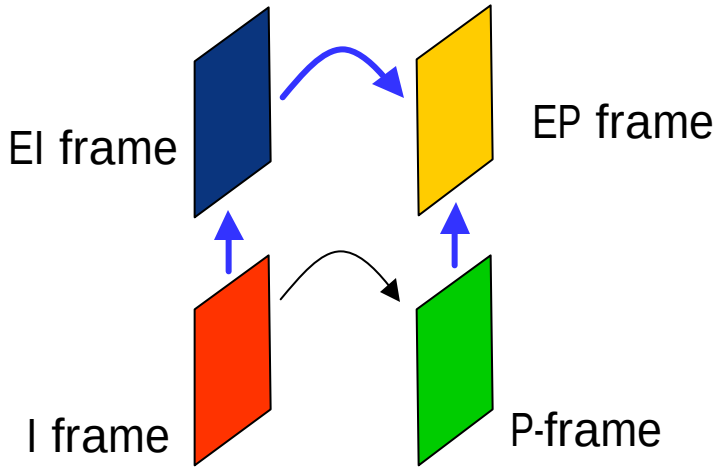
Scalable Coding: Spatial Scalability

- *Spatial scalability:* Based on refining the *spatial resolution*
  - *Base layer* is *low resolution* version of video
  - *Enh1* contains coded *difference* between upsampled base layer and original video
  - Also called: Pyramid coding



Scalable Coding: PSNR (Quality) Scalability

- *PSNR (Quality) Scalability:* Based on refining the *amplitude resolution*
  - *Base layer* uses a *coarse quantizer*
  - *Enh1* applies a *finer quantizer* to the difference between the original DCT coefficients and the coarsely quantized base layer coefficients



Note: Base & enhancement layers are at the same spatial resolution

Summary of Scalable Video Coding

- Three types of scalability coding:
  - Temporal scalability
  - Spatial scalability
  - PSNR (quality) scalability
- Scalable coding produces *different layers with prioritized importance*
- *Prioritized importance is key* for a variety of applications:
  - *Adapting* to different bandwidths, or client resources such as spatial or temporal resolution or computational power
  - *Facilitates error-resilience* by explicitly identifying most important and less important bits

Video Coding

Outline of Today's Lecture

- Motivation for video compression
- Brief review of image compression (two previous lectures)
- Video compression
  - Motion-compensated prediction
  - Motion-compensated interpolation
  - Generic (MPEG-type) video coder architecture
  - Scalable video coding
- Current video compression standards
  - What do the standards specify?
  - Frame-based video coding: MPEG-1/2/4, H.261/3
  - Object-based video coding: MPEG-4

Video Coding

Current Image and Video Compression Standards

Standard	Application	Bit Rate
JPEG	Continuous-tone still-image compression	Variable
MPEG-1	Video on digital storage media (CD-ROM)	1.5 Mb/s
MPEG-2	Digital Television	> 2 Mb/s
H.261	Video telephony and teleconferencing over ISDN	p x 64 kb/s
H.263	Video telephony over PSTN	< 33.6 kb/s
MPEG-4	Object-based coding, synthetic content, interactivity	Variable

Video Coding

Comparing Current Video Compression Standards

- *Based on the same fundamental building blocks*
  - Motion-compensated prediction and interpolation
  - 2-D Discrete Cosine Transform (DCT)
  - Color space conversion
  - Scalar quantization, runlengths, Huffman coding
- *Additional tools* added for different applications:
  - Progressive or interlaced video
  - Improved compression, error resilience, scalability, etc.
- MPEG-1/2/4, H.261/3: *Frame-based coding*
- MPEG-4: *Object-based coding* and *Synthetic video*

Video Coding

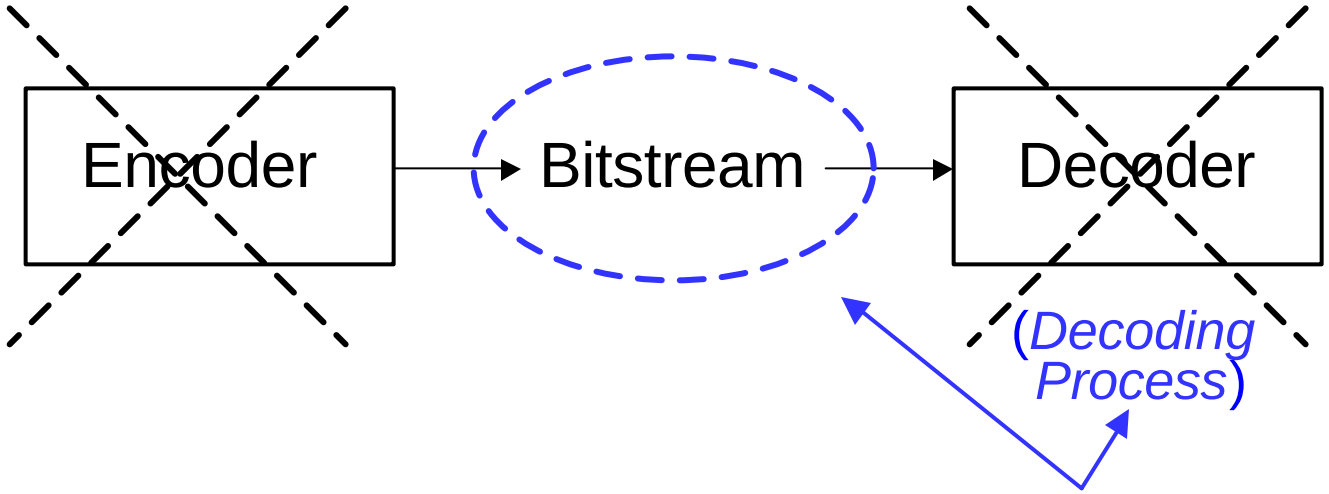
Motivation for Standards

- Goal of standards:
  - *Ensuring interoperability:* Enabling communication between devices made by different manufacturers
  - Promoting a technology or industry
  - Reducing costs

Video Coding

What do the Standards Specify?

What do the Standards Specify?



- Not the encoder
- Not the decoder
- Just the *bitstream syntax* and the *decoding process* (e.g. use IDCT, but not how to implement the IDCT)

  → Enables improved encoding & decoding strategies to be employed in a standard-compatible manner

Video Coding

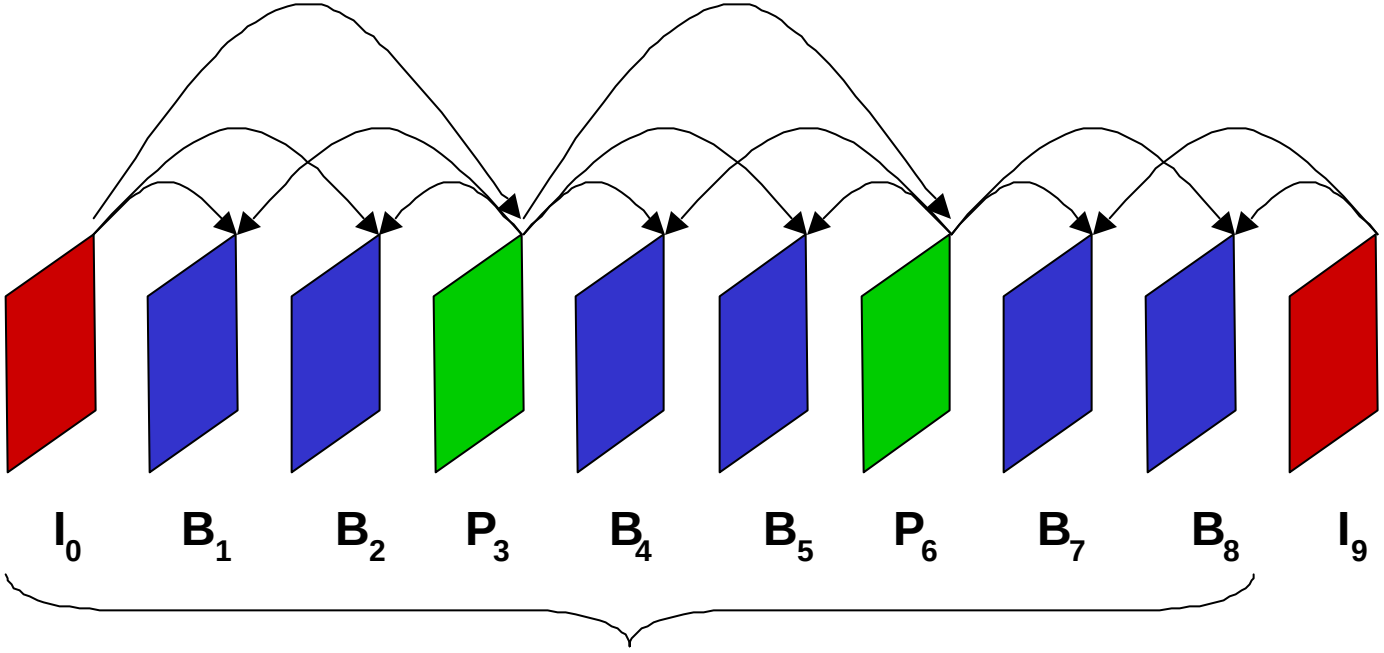
MPEG-1 and MPEG-2

- *MPEG-1* (1991)
  - Goal: Compression for digital storage media (e.g. CD-ROM)
  - Achieves VHS quality video and audio at ~1.5 Mb/s
- *MPEG-2* (1993)
  - Goal: Superset of MPEG-1 to support higher bit rates, higher resolutions, and interlaced pictures.
  - Original goal to support interlaced video from conventional television; Eventually extended to support HDTV
  - Provides: Field-based coding and scalability tools

Video Coding

MPEG Group of Pictures (GOP) Structure

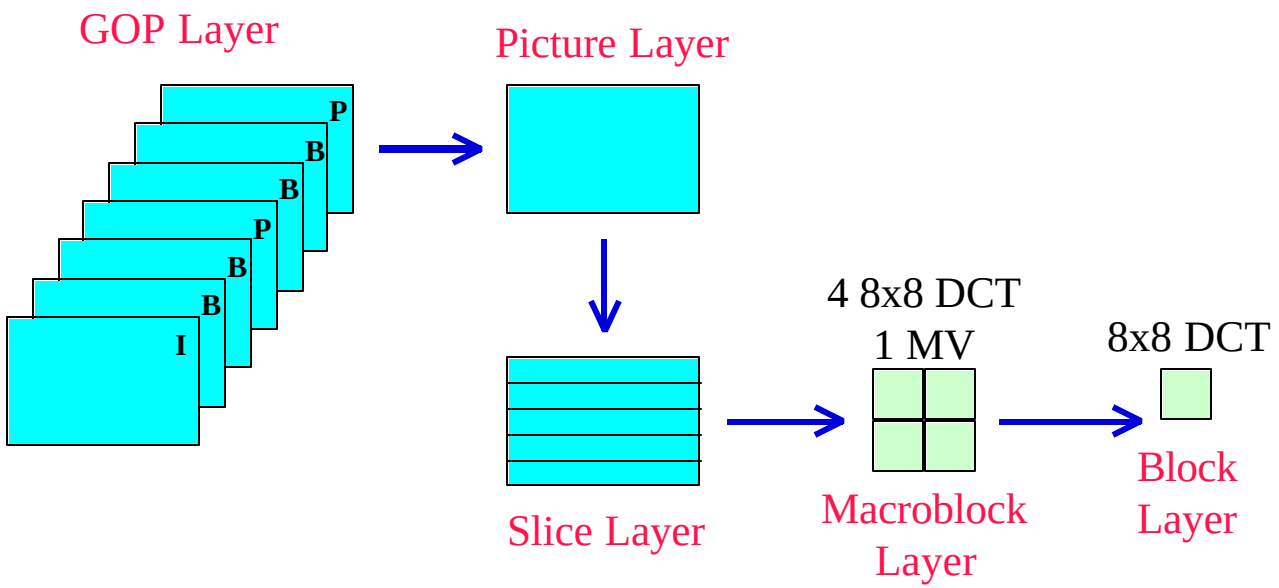
- Composed of I, P, and B frames
- Arrows show prediction dependencies
- Periodic I-frames enable random access into the coded bitstream
- Parameters: (1) Spacing between I frames, (2) number of B frames between I and P frames



MPEG GOP

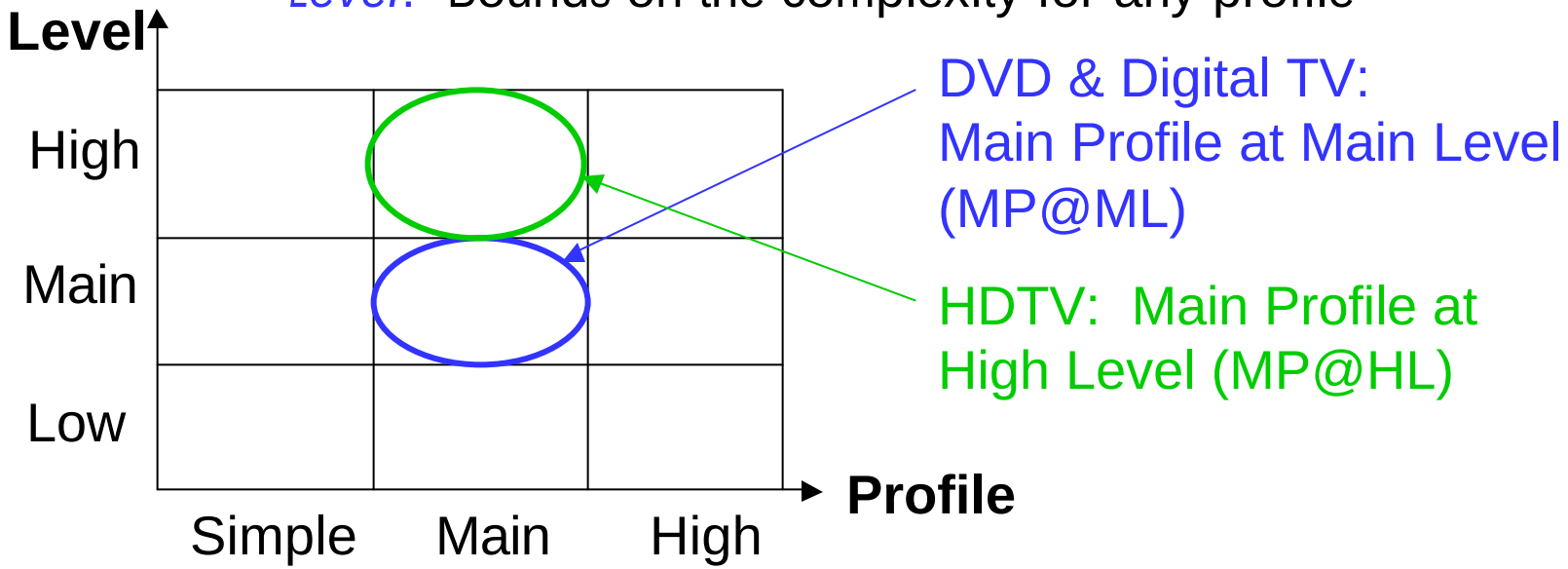
MPEG Structure

- MPEG codes video in a *hierarchy of layers*. The sequence layer is not shown.



MPEG-2 Profiles and Levels

- *Goal:* To enable more efficient implementations for different applications
  - *Profile*: Subset of the tools applicable for a family of applications
  - *Level*: Bounds on the complexity for any profile



MPEG-4

- Primary goals: *New functionalities* (not better compression)
  - *Object-based* or *content-based* representation
    - Separate coding of individual visual objects
    - *Content-based access and manipulation*
  - Integration of *natural and synthetic objects*
  - Interactivity
  - Communication over error-prone environments
- Includes frame-based coding techniques from earlier standards

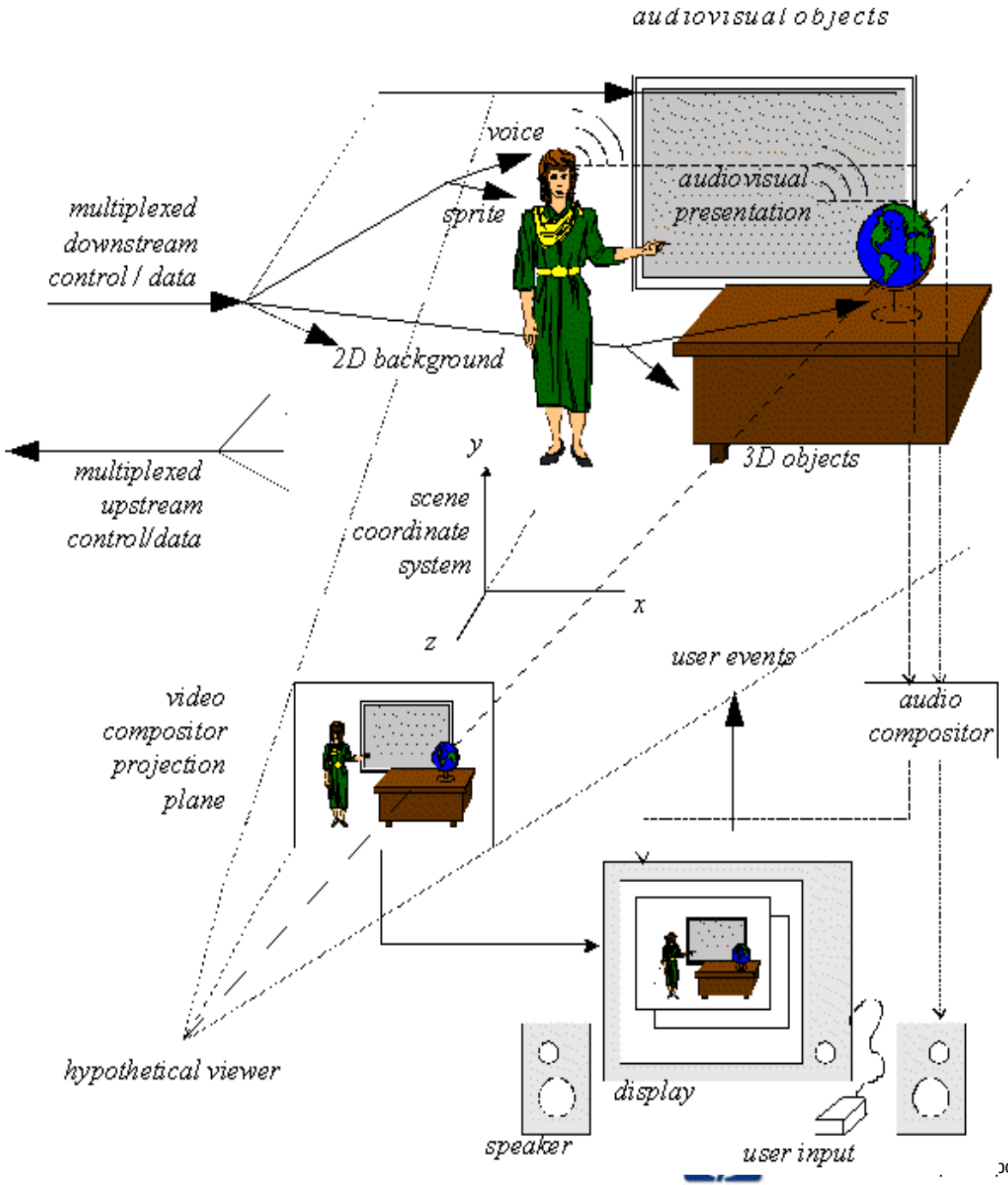
Video Coding

Comparing MPEG-1/2 and H.261/3 with MPEG-4

- *MPEG-1/2 and H.261/H.263:* Algorithms for compression
  - Basically describe a pipe for storage or transmission
  - *Frame-based*
  - *Emphasis on hardware* implementation
- *MPEG-4: Set of tools for a variety of applications*
  - Define tools and glue to put them together
  - *Object-based* and *frame-based*
  - *Emphasis on software*
  - Downloadable algorithms (not encoders or decoders)

Video Coding

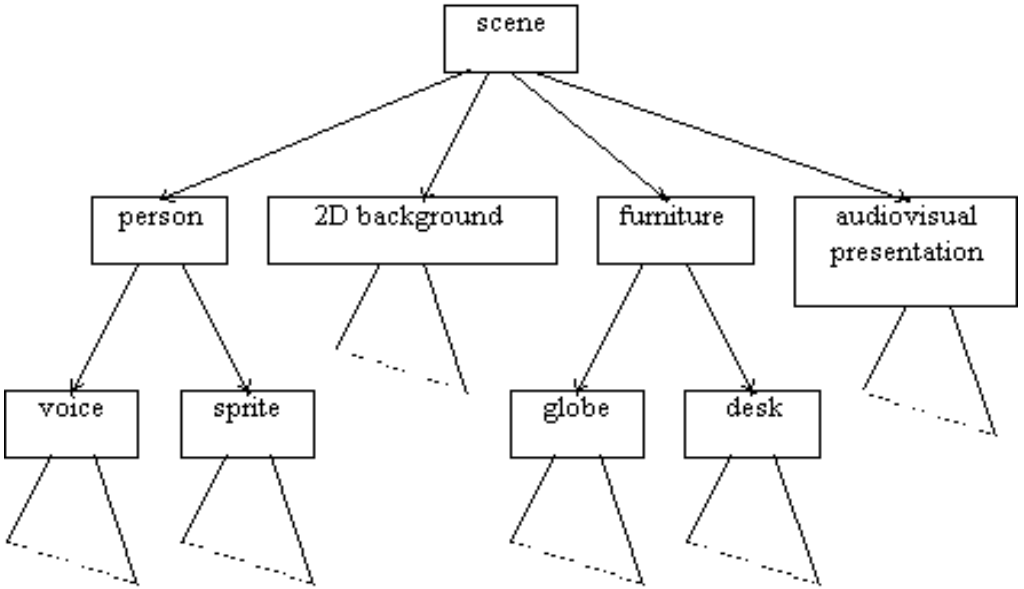
Example of MPEG-4 Scene



[MPEG Committee]

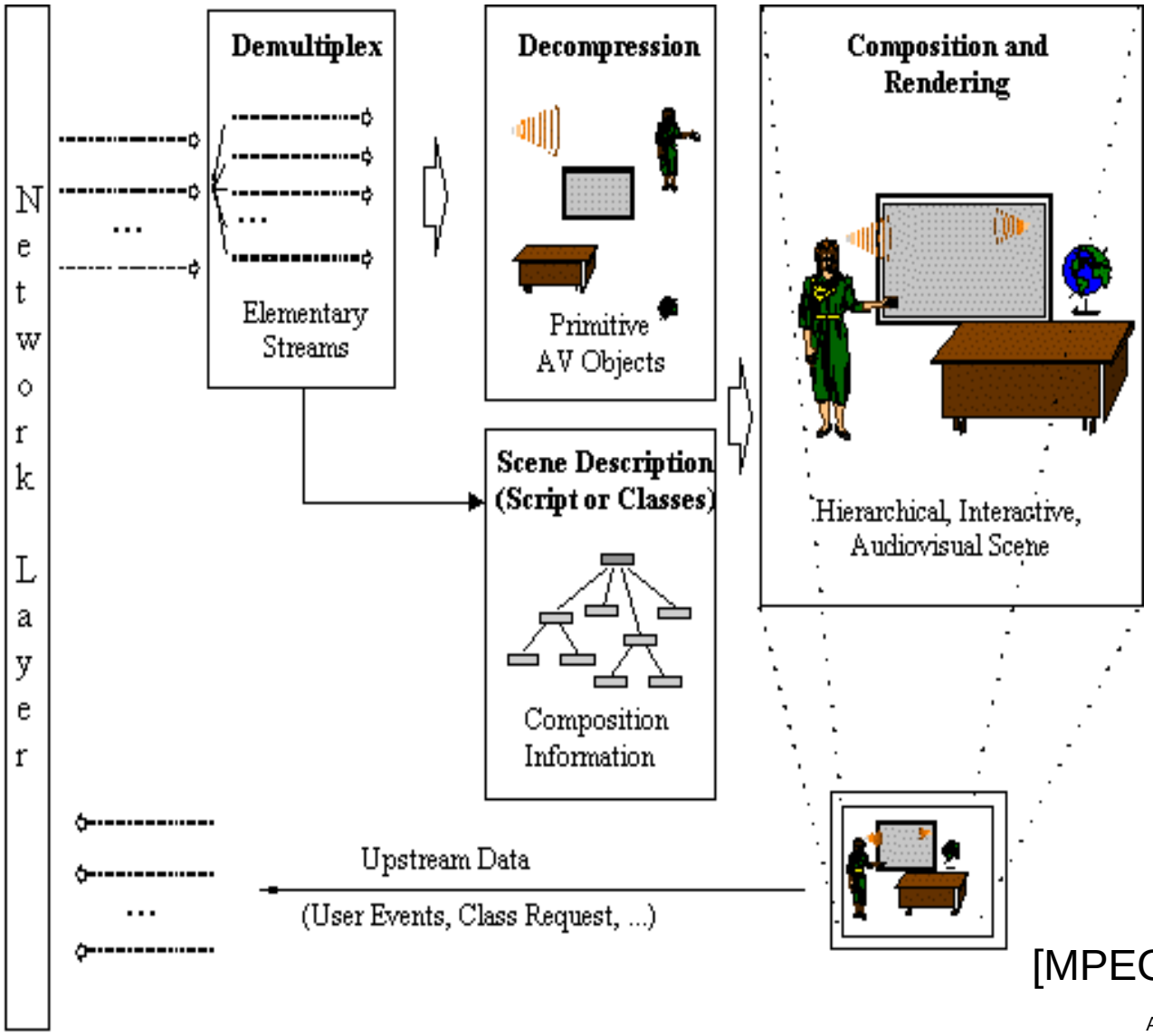
Scene Description

- Scene description:
  - Describes the *spatio-temporal positioning* of the *individual* audio & video (AV) objects to compose the scene
  - AV Objects: audio, video, natural, synthetic, 2-D, 3-D
- Hierarchical, tree structure:
  - Leaf nodes: individual AV objects
  - Other nodes: meaningful grouping



[MPEG Committee]

Example MPEG-4 Decoding Process



[MPEG Committee]

MPEG-4 Coding of Natural Video

Classes of video to represent:

Frame-based coding

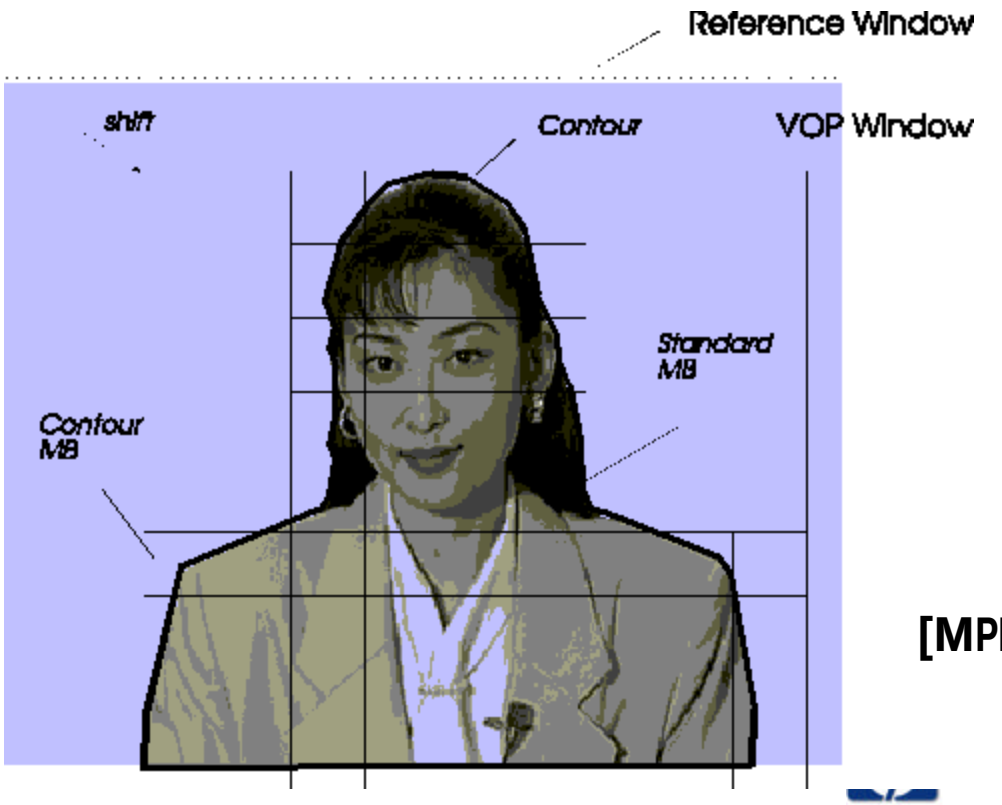
- *Rectangular images*
  - Shape (rectangle) does not change with time
  - Code motion and amplitude information
  - Use conventional coding methods, e.g. MPEG-1 / 2

Object-based coding

- *Arbitrarily shaped (non-rectangular) image regions*
  - Shape usually changes with time
  - Must code *motion, amplitude (texture) and shape*
    - → Arbitrary & time-varying shape complicates coding
  - Also describe how objects are *composed* to form scene (scene description)
  - *Separate* encoding and decode of each object

Example of Arbitrarily Shaped Object

- Arbitrarily shaped 2-D object (image region):
  - *Video object plane (VOP)* in MPEG-4



[MPEG Committee]

Video Coding

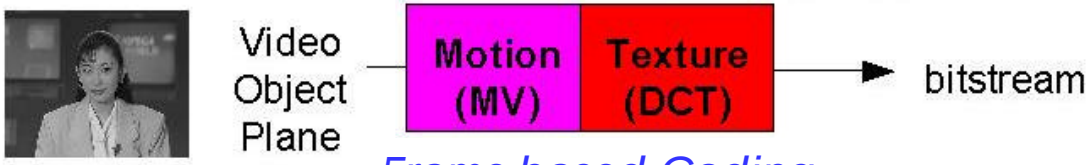
Comments on Segmentation

- *Segmentation* of video into objects is *not standardized* (part of encoder)
- Different segmentations scenarios:
  - Sometimes segmentation is available, e.g. synthetically generated content
  - Sometimes it is relatively easy, e.g. bluescreening or video-conferencing
  - Usually it is *very difficult*

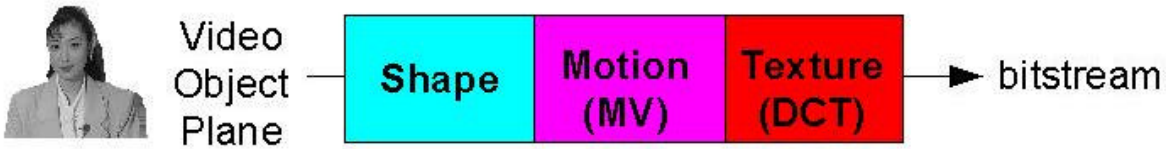
MPEG-4 Natural Video Coding

- Extension of MPEG-1 / 2-type algorithms to code arbitrarily shaped objects

  → Based on *Block-DCT and Block-ME / MC-P*



Frame-based Coding



Object-based Coding

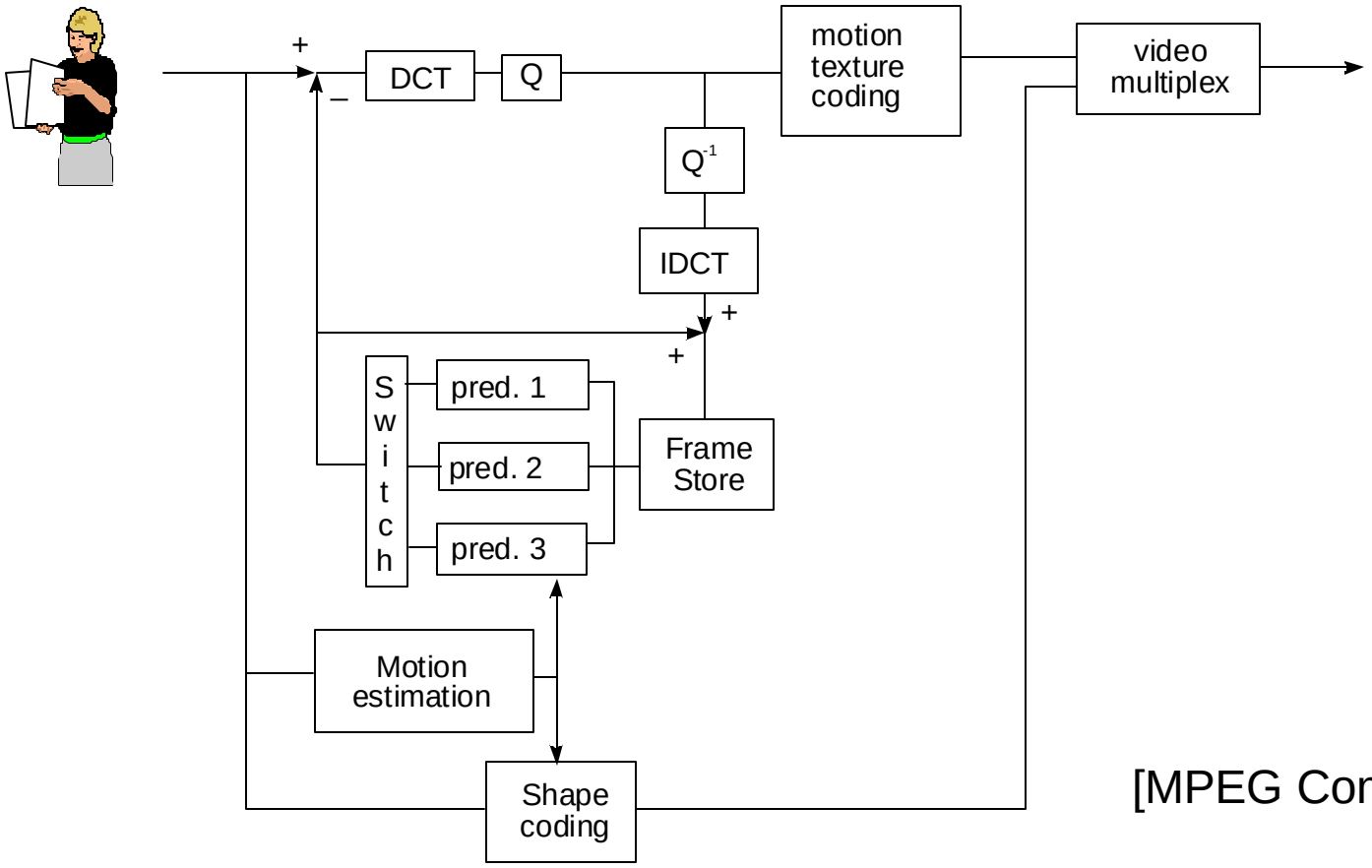
[MPEG Committee]

Video Coding

MPEG-4 Object-based Coding

- *Code amplitude (texture), shape, and motion*
  - Both texture and shape undergo (different) motion!
- Brief coverage today:
  - Texture coding for arbitrarily shaped objects
    - Intra-frame (still-frame)
    - Inter-frame (motion-compensated prediction)
  - Shape coding for arbitrarily shaped objects
    - Intra-frame and Inter-frame
  - Sprites (background object mosaic)
- MPEG-4 algorithms are extensions of MPEG-1/2-type algorithms → *Based on Block-DCT and Block-ME/MC-P*

Overview of MPEG-4 Encoder for Encoding an Arbitrarily Shaped Object



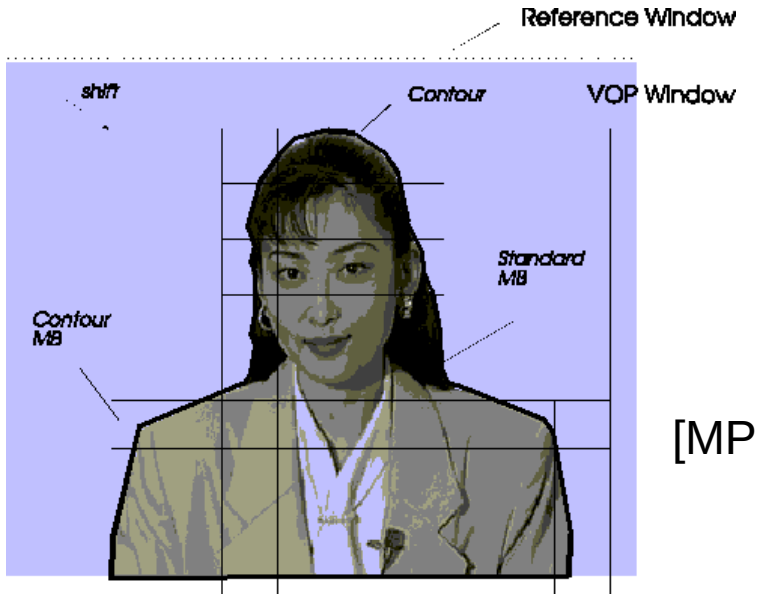
[MPEG Committee]

Note: The spirit of this figure is correct, but the specifics are not

Video Coding

Coding the Texture of an Arbitrarily Shaped Object

- Texture (amplitude) coded by Block-DCT adapted for arbitrarily shaped support
  - 1) Embed VOP in rectangle
  - 2) Separate processing of each 8x8 block
    - a) Interior → Conventional Block-DCT
    - b) Exterior → Discard
    - c) Boundary → Extrapolate then Block-DCT

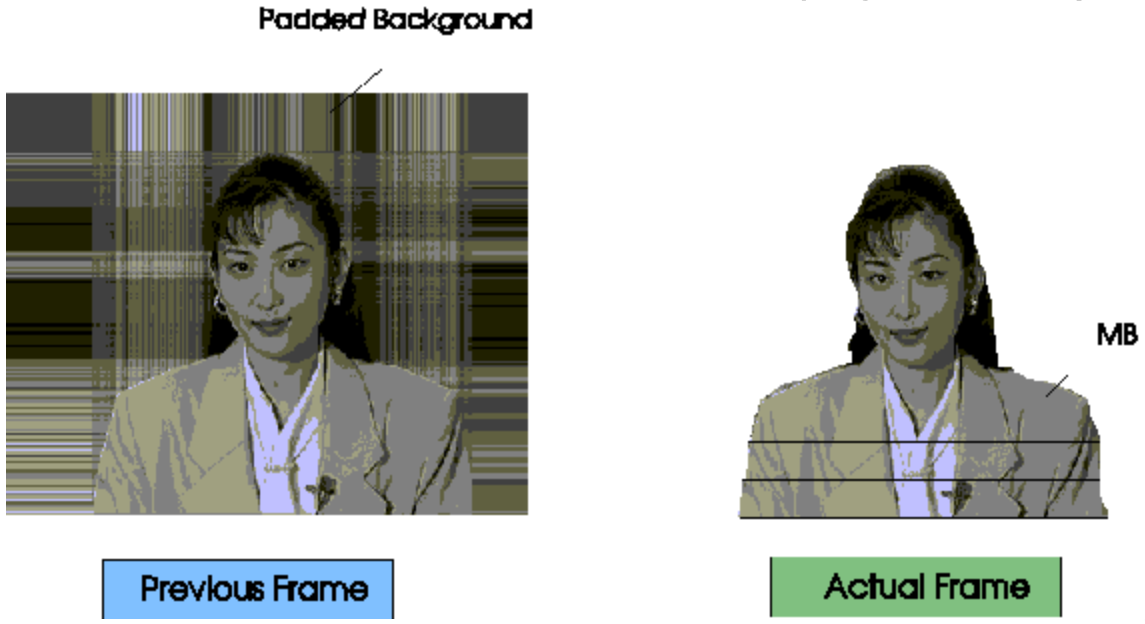


[MPEG Committee]

Video Coding

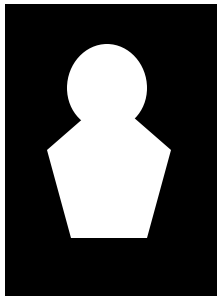
MC-Prediction for Texture Coding of Arbitrarily Shaped Object

- Block-based ME/MC-P adapted for arbitrarily shaped support:
  1) Extrapolate arbitrarily shaped object to fill rectangle
  2) Perform conventional block-based ME/MC-P
     - Error metric computed only over object's support in current frame
- Also: Parametric motion models (e.g. affine, perspective)



[MPEG Committee]

Binary Shape Coding



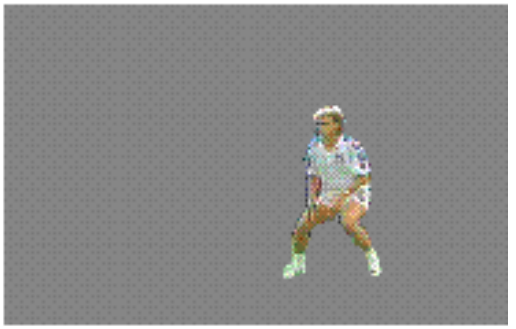
Binary shape (object support is white)

- Opaque objects: Each pixel either inside or outside support
  - Shape given by *binary alpha map* (bitmap or binary mask)
- Many possible approaches for lossless and lossy shape coding
  e.g. Describe shape by chain code, polynomials, splines, bitmap
- MPEG-4: Block-based Context-based Arithmetic Coding (CAE)
  1) Embed support in rectangle
  2) Separate processing of 16x16 blocks
     a) Opaque blocks (completely within object)
     b) Transparent blocks (completely outside object)
     c) Boundary blocks $\rightarrow$ CAE
- Also motion compensated CAE

Sprite Coding (Background Prediction)

- *Sprite: Large background image*
  - Hypothesis: Same background exists for many frames, changes resulting from camera motion and occlusions
- One possible coding strategy:
  1. Code/transmit entire sprite once
  2. Only transmit camera motion parameters for each subsequante frame
- Significant coding gain for some scenes

Sprite Coding Example



Sprite (background)

Foreground Object

Reconstructed Frame

[MPEG Committee]

Video Coding

Related MPEG Standards (non-compression)

- MPEG-7 "Multimedia Content Description Interface"
  - Goal: A method for describing multimedia content to enable efficient searching and management of multimedia.
- MPEG-21 "Multimedia Framework"
  - Goal: To enable the electronic commerce of digital media content.

Video Coding

Review of Today's Lecture

- Motivation for video compression
- Brief review of image compression (two previous lectures)
- Video compression
  - Motion-compensated prediction
  - Motion-compensated interpolation
  - Generic (MPEG-type) video coder architecture
  - Scalable video coding
- Current video compression standards
  - What do the standards specify?
  - Frame-based video coding: MPEG-1/2/4, H.261/3
  - Object-based video coding: MPEG-4
- *Next lecture:* Compressed-Domain Video Processing
  → Efficient processing of compressed video

References and Further Reading

General Video Compression References:

- J.G. Apostolopoulos and S.J. Wee, ``Video Compression Standards'', *Wiley Encyclopedia of Electrical and Electronics Engineering*, John Wiley & Sons, Inc., New York, 1999.
- V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*, Boston, Massachusetts: Kluwer Academic Publishers, 1997.
- J.L. Mitchell, W.B. Pennebaker, C.E. Fogg, and D.J. LeGall, *MPEG Video Compression Standard*, New York: Chapman & Hall, 1997.
- B.G. Haskell, A. Puri, A.N. Netravali, *Digital Video: An Introduction to MPEG-2,* Kluwer Academic Publishers, Boston, 1997.

MPEG web site:

http://drogo.cselt.stet.it/mpeg

References and Further Reading (cont.)

Video Compression Standards Documents

- *Video codec for audiovisual services at px64 kbits/s, ITU-T Recommendation H.261,* International Telecommunication Union, 1990.
- *Video coding for low bit rate communication, ITU-T Recommendation H.263,* International Telecommunication Union, version 1, 1996; version 2, 1997.
- *ISO/IEC 11172, Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbits/s.* International Organization for Standardization (ISO), 1993.
- *ISO/IEC 13818. Generic coding of moving pictures and associated audio information.* International Organization for Standardization (ISO), 1996.
- *ISO/IEC 14496. Coding of audio-visual objects.* International Organization for Standardization (ISO), 1999.