Department of Computer Science
National Tsing Hua University

CS 5263: Wireless Multimedia Networking Technologies and Applications

Basics of Video Coding

Instructor: Cheng-Hsin Hsu

Acknowledgement: The instructor thanks Prof. Mohamed Hefeeda at Simon Fraser University for sharing his course materials

Outline

- Video Coding Standard
- Intra and Inter-frame Compression
- MPEG Video Compression
- Scalable Video Coding
- Error Propagation
- Introduction to H.264/AVC

Video

- Enabled by two properties of human vision system
- Persistence of vision:
  - Tendency to continue to see something for a short period after it is gone
- Flicker fusion:
  - The ability of human vision system to fuse successive images into one fluid moving image
- Interesting discussion on video capturing, editing, processing, standards, etc. in [Ch. 6, 7 of Burg09]
  - We focus on video coding/compression

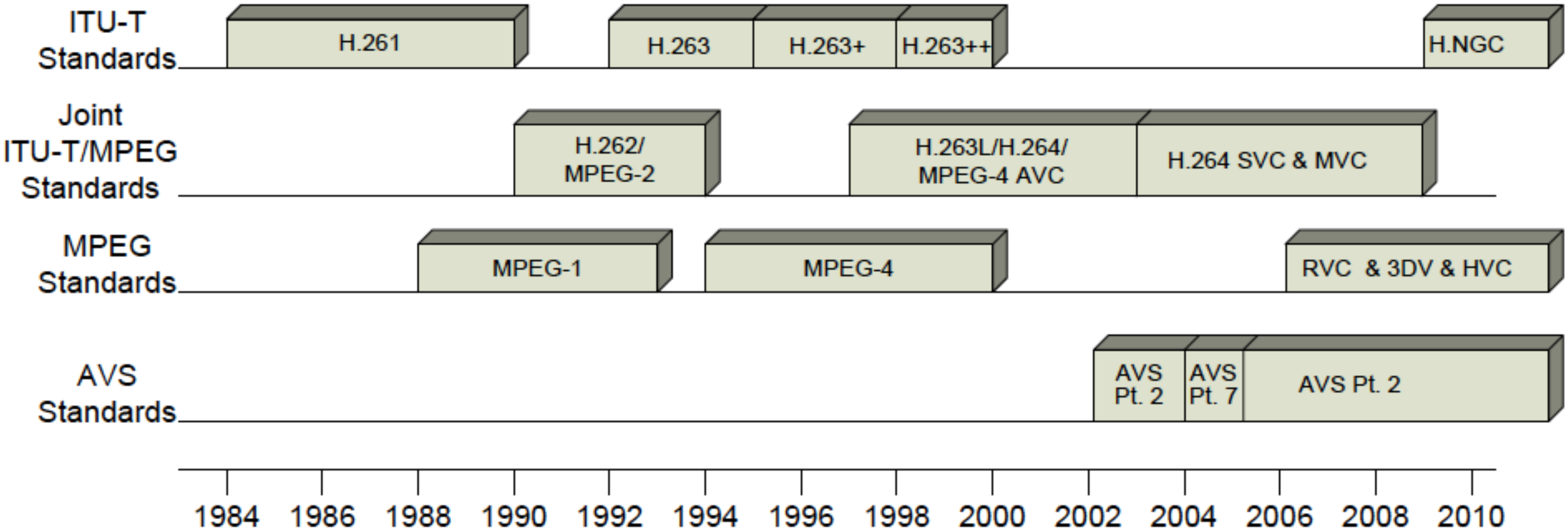
Video Coding Standards

- ITU: International Telecommunication Union
  - Coordinates global telecom networks and services
  - 3 sectors: ITU-T (telecom); ITU-R (radio); ITU-D (development)
  - Video Coding Expert Group (subcommittee for ITU-T) produced the H.26* codec series, where * is replaced by a digit, e.g., 1, 2, 3, 4
  - H.261 (~1990): for video conferencing and video over ISDN
  - H.263 (~1995): improved H.261 to transmit video over phone lines
  - H.264/AVC (~2003): most common, high compression ratios
    - Also known as MPEG-4 Part 10 or MPEG-4/AVC
- NOTE: Most video coding standards only specify decoder operation and stream syntax → lots of room for innovation at the encoder side

Video Coding Standards

- **MPEG: Motion Picture Expert Group**
  - From ISO (International Organization for Standardization) and IEC (International Electrotechnical Commission)
  - Audio and video coding for a wide range of multimedia applications
  - Several versions: MPEG-1, -2, -4, (-7 & -21: for content description)
    - MPEG-1 = ISO/IEC 11172 Standard
  - Each version has several Parts; each has multiple Profiles
  - Each Part is related to specific component: audio, video, system, etc
    - MPEG-1 has 6 parts, MPEG-2 has 9, and MPEG-4 has 23
    - MPEG-4 Part 10: Advanced Video coding = H.264/AVC
  - Each Profile identifies subset of features to be implemented, e.g., sub-sampling type, quantization methods, etc
  - Each Profile can have multiple levels
    - Level indicates encoding computation complexity

Timeline for Video Coding Standards



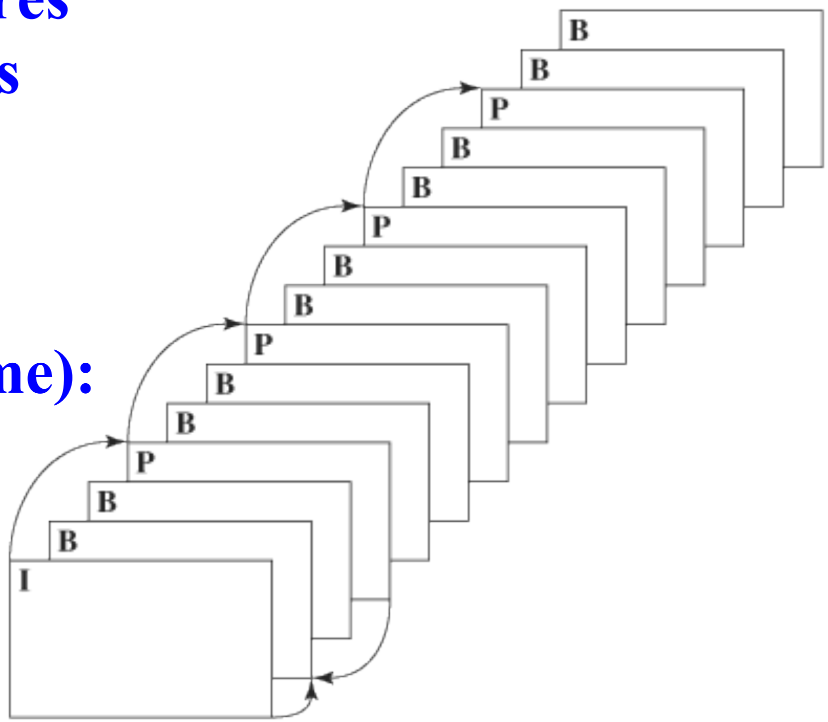
J. Dong and K. Ngan, Present and Future Video Coding Standards, book chapter at http://jdong.h265.net/2010_chapter.pdf

Video Coding

- **Video is a sequence of images**
  - **Typical frame rates (fps): 29.97, 24, 15**
- **Common method for compressing videos**
  - **Remove redundancies inside the frame**
    - **Intraframe compression or spatial compression**
    - **Usually using transform encoding (e.g., in JPEG)**
  - **Remove redundancies across frames**
    - **Interframe compression or temporal compression**
    - **Visual scenes do not change much in neighboring frames →**
    - **Detect how objects move from one frame to another**
      - motion vector; computed by motion estimation
    - **Use motion vectors and differences among frames**
      - Differences are small → good for compression

MPEG Compression: Basics

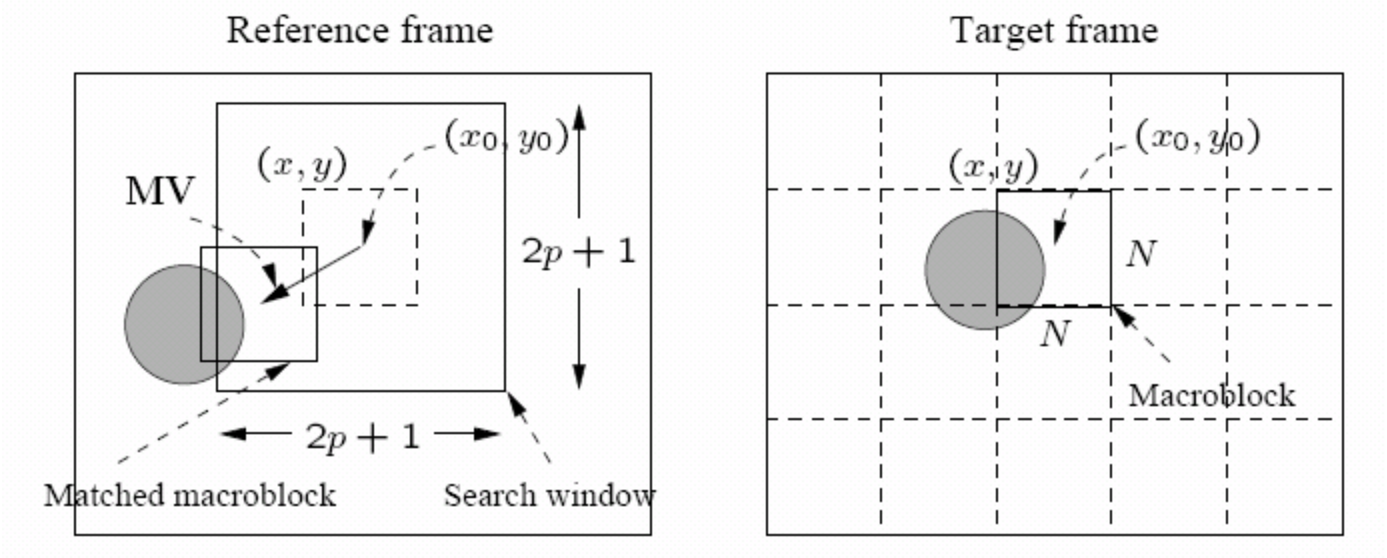
- Divide video into groups of pictures (GOPs), identifying I, P, B frames
- I frame (intraframe):
  - Compressed spatially with JPEG
- P frame (forward prediction frame):
  - Compressed temporally relative to a preceding I or P frame
  - Then compressed spatially
- B frame (bidirectional frame):
  - Compressed temporally relative to preceding and following I and/or P frames
  - Then compressed spatially



Spatial Compression

- Frames are divided into 16 x 16 macroblocks
- Chroma subsampling is usually used, e.g., 4:1:1, 4:2:0
- → We get 8 x 8 blocks for Y and CbCr
- Then we apply DCT and the rest of JPEG compression

Temporal Compression: Motion Compensation



- **Motion compensation is performed at macroblock level**
- **Current frame is referred to as *Target Frame***
- **Closet match is found in previous and/or future frame(s), called *Reference Frame(s)***
  - **Search is usually limited to small neighborhood: range [−*p*, *p*] ➔ search window size: (2*p* + 1) x (2*p* + 1)**
  - **Displacement is called *Motion Vector* (MV)**

Search for Motion Vectors

- **Difference between two macroblocks can be measured by Mean Absolute Difference (MAD):**

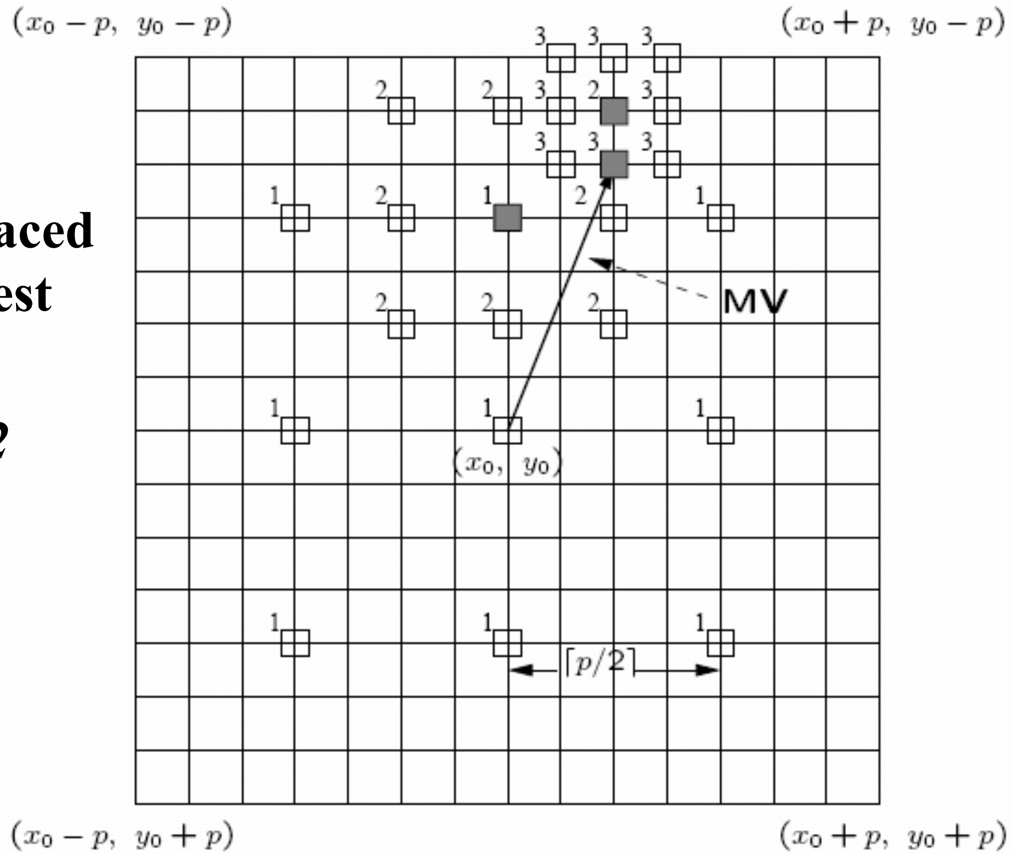
$$MAD(i,j) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\left|T(x+k, y+l) - R(x+i+k, y+j+l)\right|$$

  - ***N:*** **size of macroblock**
  - ***k, l:*** **indices for pixels in macroblock**
  - ***i, j:*** **horizontal and vertical displacements**
  - **$T(x+k, y+l)$: pixels in macroblock in Target frame**
  - **$R(x+i+k, y+j+l)$: pixels in macroblock in Reference frame**

- **We want to find the MV with minimum MAD ← Why?**

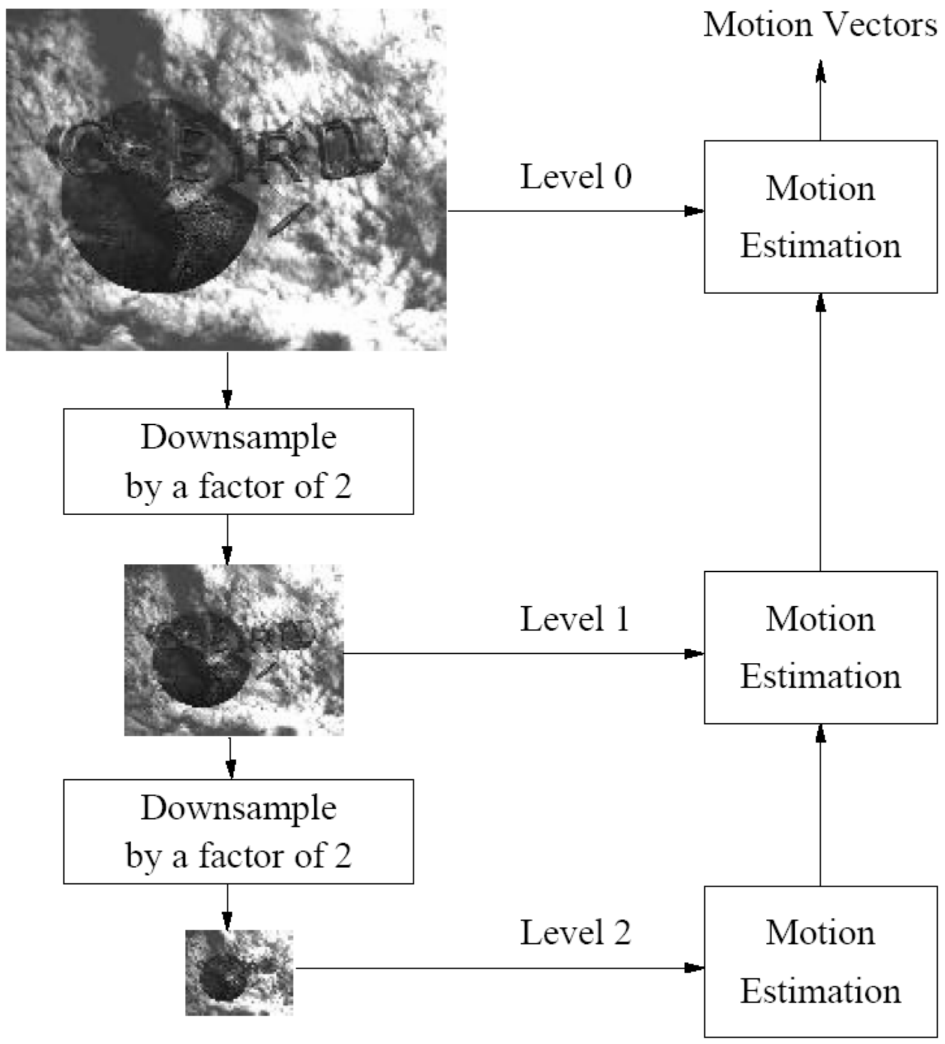
Search for Motion Vectors

- Full search: check the (2p + 1) x (2p + 1) window ← expensive
  ➔ Time complexity: $O\ (p^2\ N^2)$ ← conservative: sub-pixel, multiple-reference
- 2D Logarithmic search
  - Compare 9 blocks evenly spaced within distance *p* and find best matching macroblock
  - Compare 9 blocks within *p/2*
  - And so on
  - Complexity: $O\ (log(p)\ N^2)$

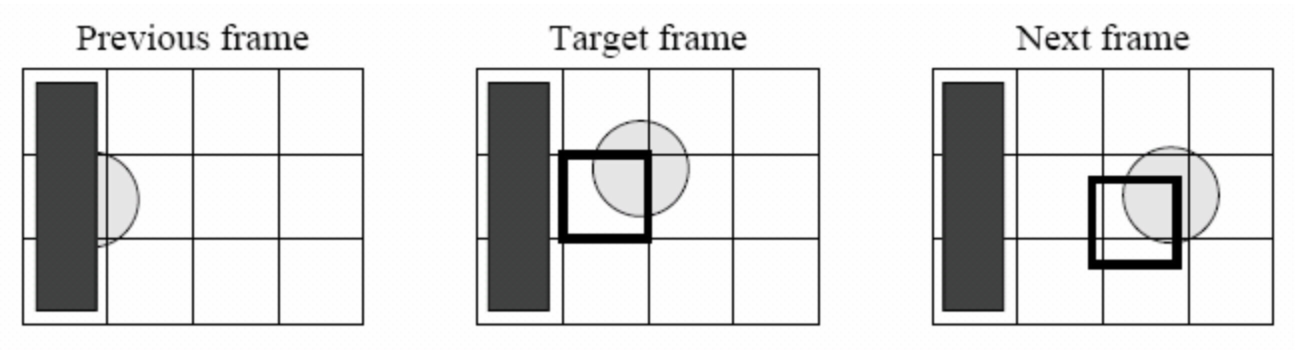


Search for Motion Vectors

- **Hierarchical search**
  - **multi-resolutions by down sampling**
  - **Search for MV in reduced resolutions first (faster)**

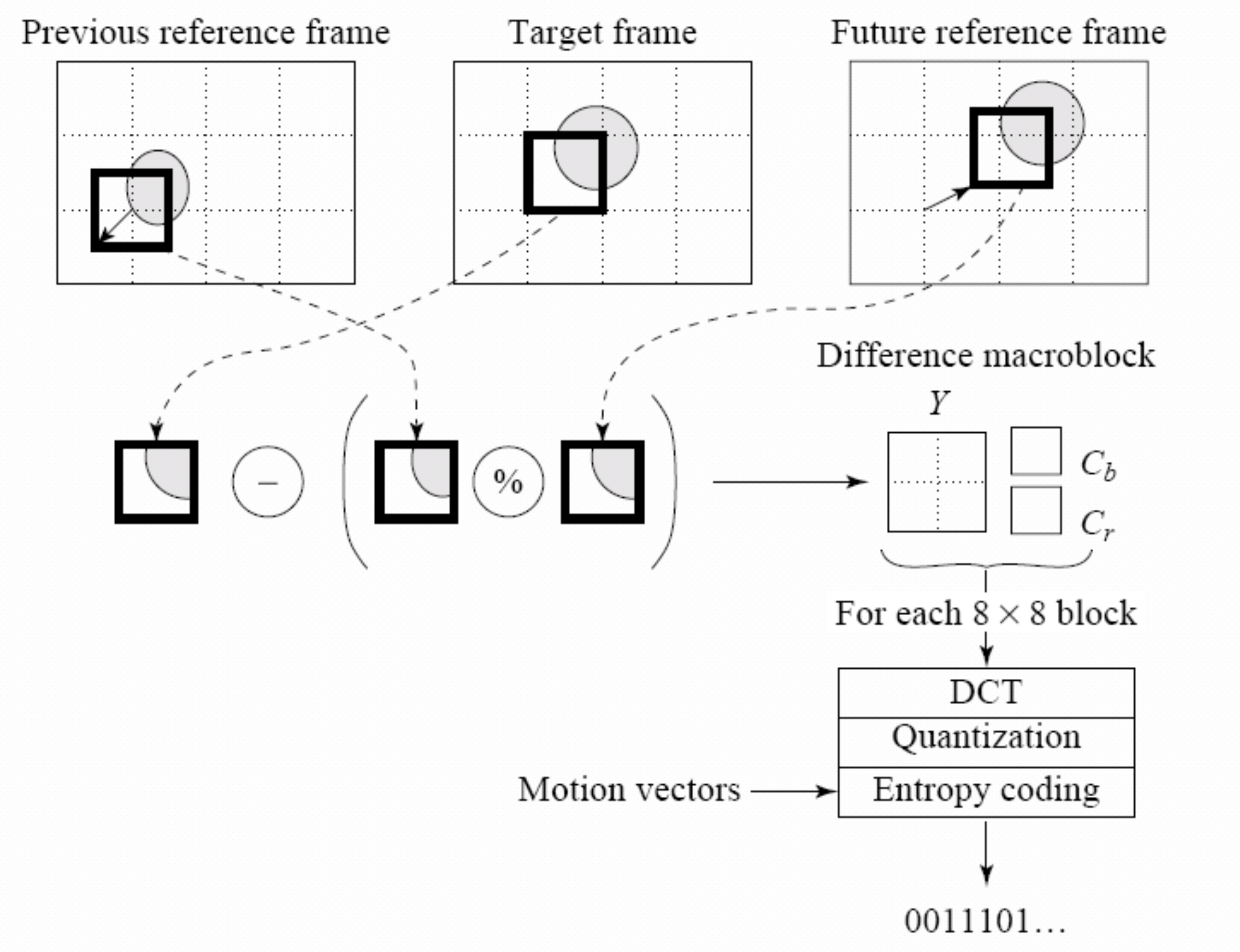


The Need for B Frame

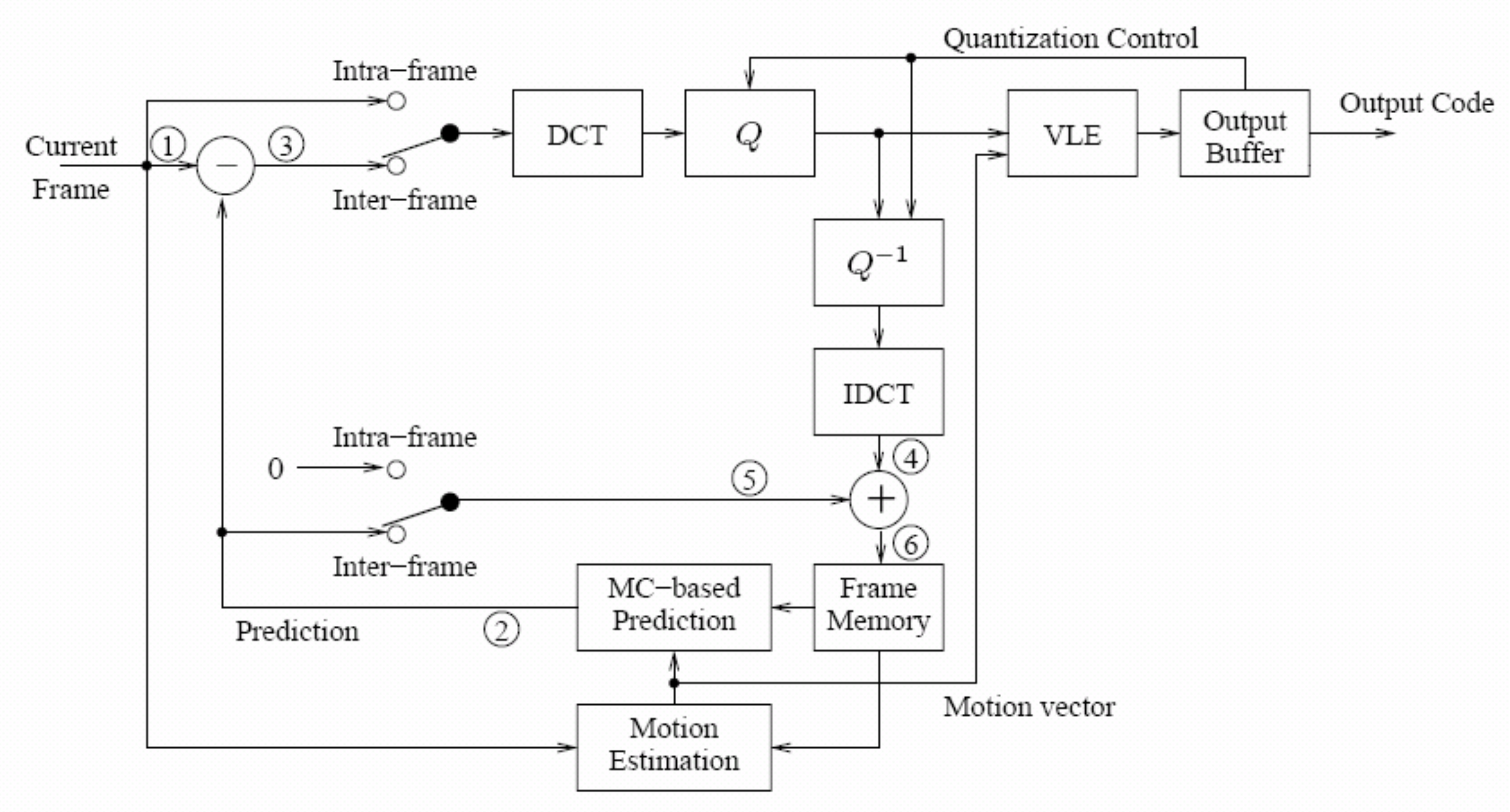


- Sometimes a good match cannot be found in previous frame, but easily found in next frame
- B-frames can use one or two motion vectors
  - 1 MV: if either previous or next frame is used
  - 2 MVs: if both frames are used
    - Difference is computed between target frame and the average of both previous & next frames ← weighted sum is also possible

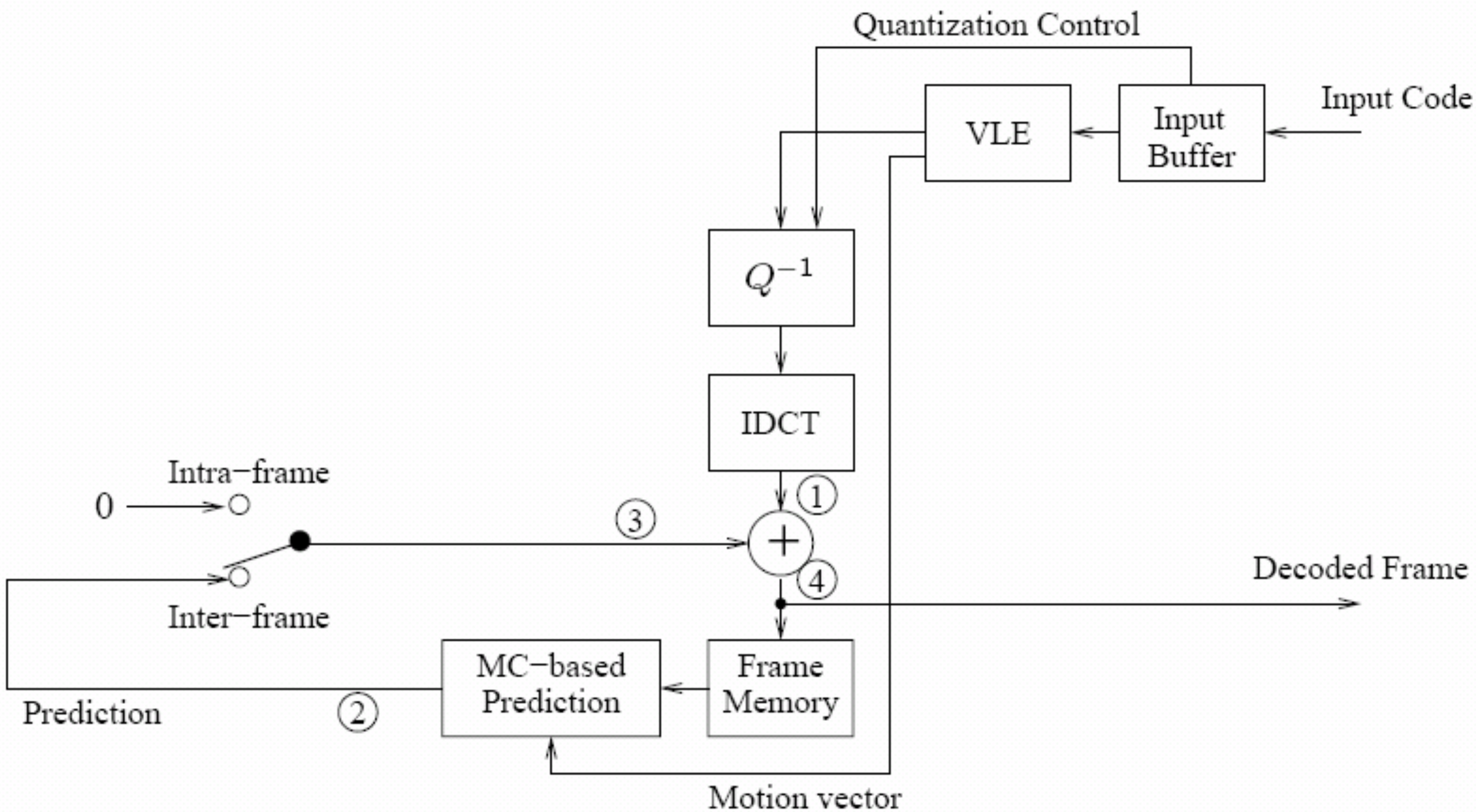
B-frame coding



Basic Video Encoder

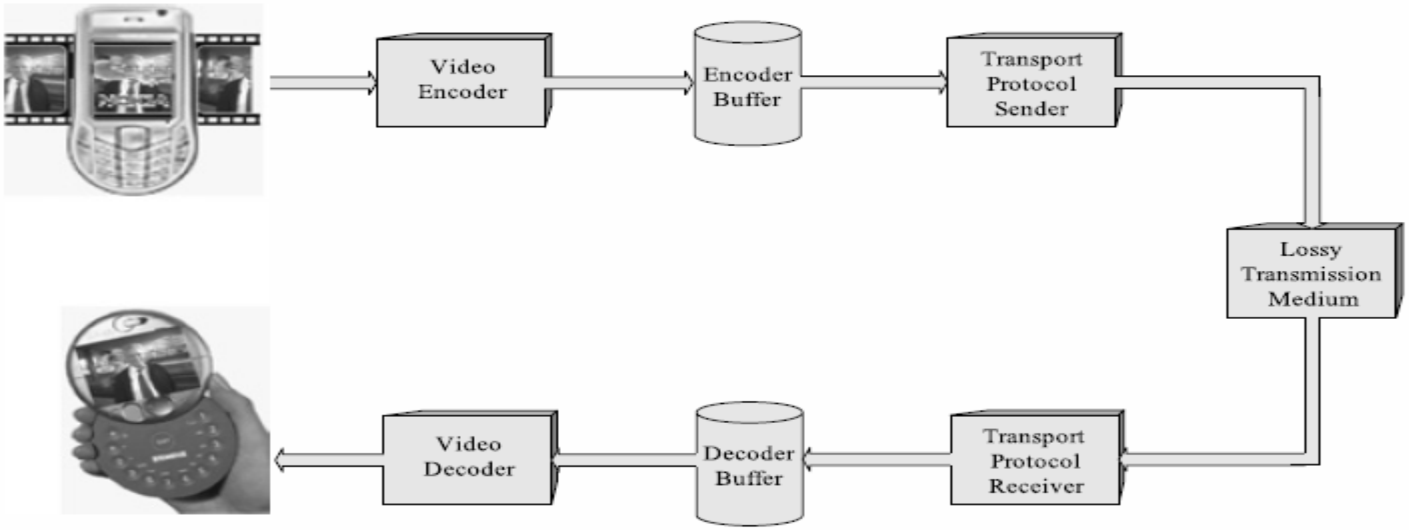


Basic Video Decoder



Errors and Video Coding Tools

- Video is usually transmitted over lossy channels, e.g., wireless network and the Internet
  - ➔ some video data may not arrive at the receiver
- Video data is also considered lost if it arrives late



Error Propagation

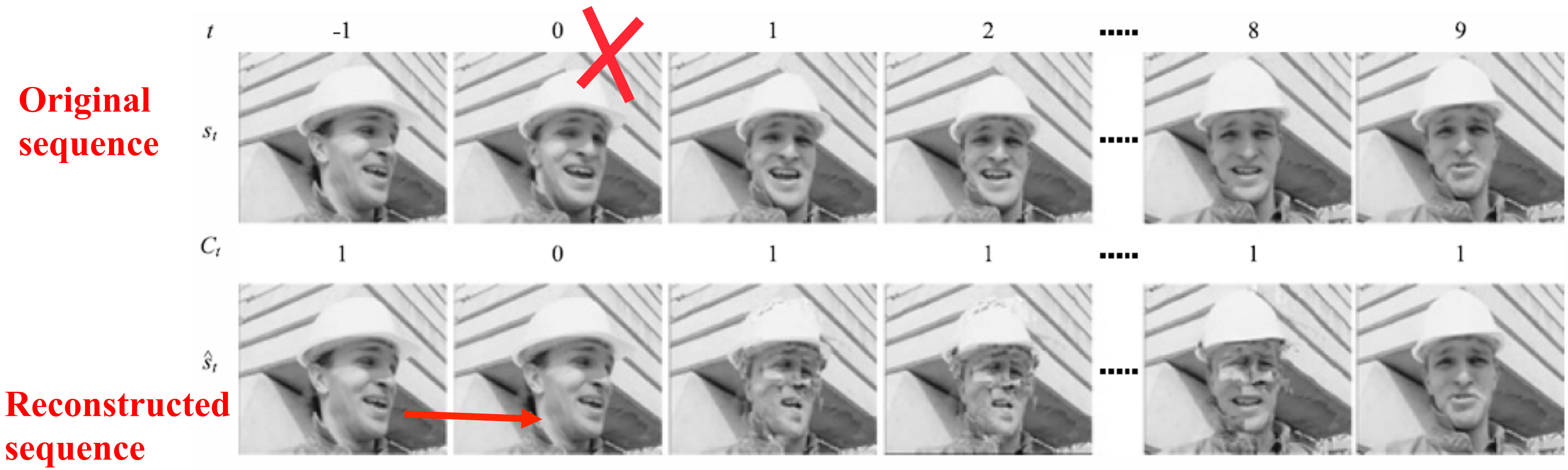
- Because of temporal compression, decoding of a video frame may depend on a previous frame
  - ➔ Thus, even if a frame arrives on time, it may not be decoded correctly
- Example: to decode a P-frame, we need its reference frame (previous I or P-frame in the GoP)
- What do we do if the reference frame is lost/damaged?
- Try to conceal the lost frame
- Simplest approach is:
  - Decoder skips decoding the lost frame and the display buffer is not updated

Error Propagation

- Problems with the simple error concealment approach?
- Some loss in the smoothness of the motion
  - We display the same frame for twice its time
- And ….
- some future frames could be damaged (error propagation). Why?
- Because the decoder will use a different reference frame in decoding than one used during encoding
  - Known as reconstruction mismatch

Error Propagation: Example

- Frame at t = 0 is lost ➔ use frame at t = -1 as reference for decoding frames at t = 1, 2, …, 8 ➔
- Frames 1—8 are not perfectly reconstructed, although they were received correctly ➔ error propagated through all of them
- Error stooped with an new intra-coded (I) frame (at t = 9)



Handling Error

- Note: Most error-resilience coding tools (e.g., using intra-coded macro blocks, slice coding, …) decrease compression efficiency
  - Trade off: error resiliency ←→ compression efficiency
- Shannon's Separation Principle
  - Separate compression (source coding) from transport (channel coding), that is:
  - Employ link features (e.g., retransmission, FEC) to avoid/ recover from losses, and
  - Maintain high compression efficiency (without worrying about error resiliency)
- In many practical situations, we cannot totally avoid losses or recover from them, e.g., in low delay systems

Handling Error: Principles

- Loss correction <u>below</u> codec layer
  - Done in transport layer (i.e., coder is unaware of it) to minimize amount of losses
  - Examples:
    - TCP (retransmission)
    - Forward Error Correction (FEC)
- Error detection
  - Detect and localize erroneous video data
- Prioritization methods
  - If losses are unavoidable, minimize losses for important data (e.g., motion vectors, reference frames)

Handling Error: Principles (cont'd)

- **Error recovery and concealment**
  - In case of losses, minimize visual impacts
- **Encoder-decoder mismatch avoidance**
  - Reduce encoder-decoder mismatch to reduce error propagation
- **Details are given [Ch 2, SC07]**

Coding Tools for Error Resilience

- **We discuss coding tools to improve error resiliency**
  - **These are used for both compression AND error resiliency**
  - **E.g., Slice coding, Flexible Reference Frame, …**
- **Different tools are used in different standards and in different profiles within the same standard**
- **Most tools are included in the state-of-the-art MPEG-4/AVC (aka H.264/AVC) video coding standard**

Slice Structured Coding

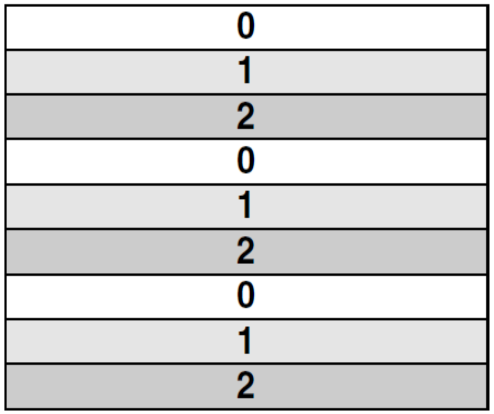
- **A video frame is divided in multiple slices**
  - Different parts of the frame have different visual complexities (i.e., data redundancies)
  - A whole frame may be larger than a network packet size ➔ lower layers may divide it into multiple packets (and losing any of them makes decoding difficult)
- **Slice: group of macroblocks (an independent unit)**
  - Encoded and decoded independently of others
  - intra prediction and motion vector prediction are not allowed across slices
  - Has slice header
  - Could have fixed number of blocks or
  - fixed number of bits (preferred for network transmission)

Slice Structured Coding: Example

- Mapping of blocks to slices in raster-scan manner
  - Simple (used in MPEG-1, MPEG-2, and H.263)
- Any disadvantages of this simple mapping?
- Not very efficient in error concealment: if MB is damaged, neighbouring blocks are used to estimate it
- Does not support Region of Interest (ROI) protection:
  - Video conference: Human face in frame → higher protection against error ← Solution: Slice Group

Flexible Macroblock Orderg in (FMO) in AVC

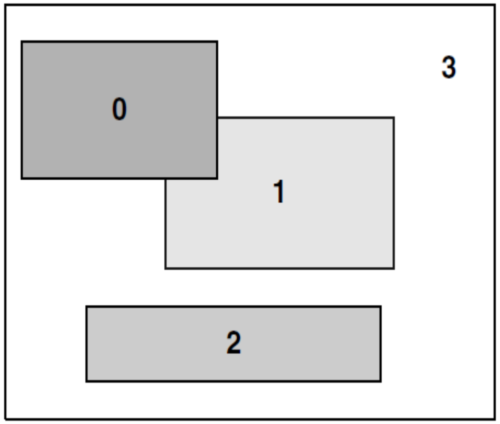
Interleaved



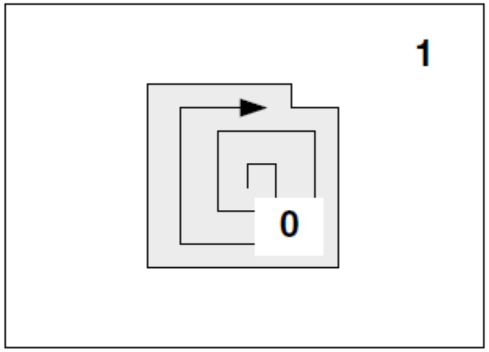
Dispersed

0	1	2	3	0	1	2	3	0	1	2
2	3	0	1	2	3	0	1	2	3	0
0	1	2	3	0	1	2	3	0	1	2
2	3	0	1	2	3	0	1	2	3	0
0	1	2	3	0	1	2	3	0	1	2
2	3	0	1	2	3	0	1	2	3	0
0	1	2	3	0	1	2	3	0	1	2
2	3	0	1	2	3	0	1	2	3	0
0	1	2	3	0	1	2	3	0	1	2

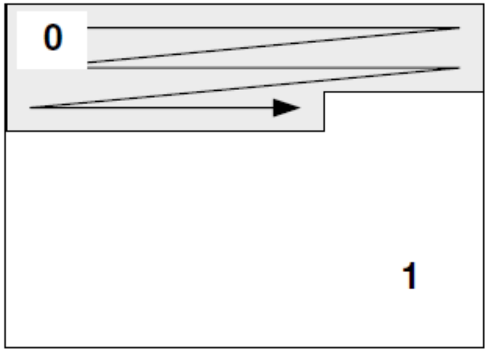
Foreground and Background



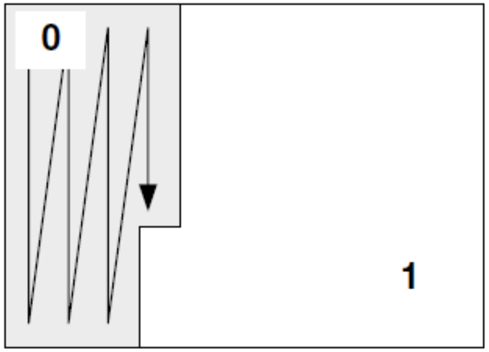
Box-out



Raster



Wipe



ASO in H.264/AVC

- Arbitrary Slice Order (ASO) allows
  - sending/receiving slices of frame in any order relative to each other

- ASO can improve end-to-end delay in real-time applications
  - For example, sending the ROI slices first
  - particularly in networks having out-of-order delivery behavior

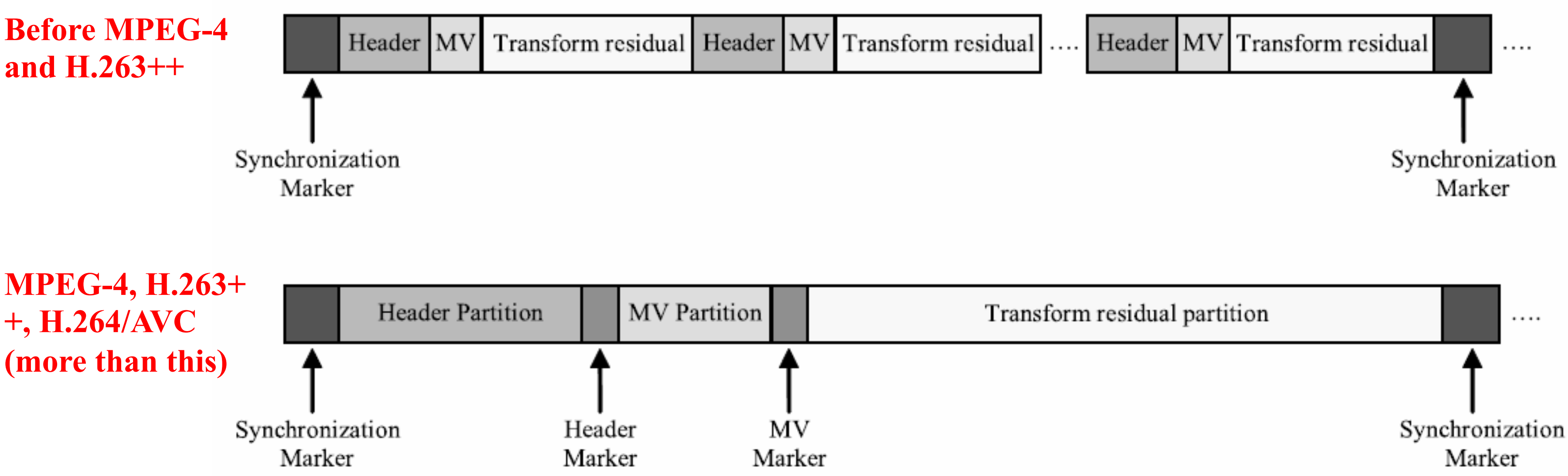
Scalability

- **Supports decoding at different extraction points with a single coded stream**
  - **<Resolution, Frame Rate, QP>: {<VGA, 15 fps, 12>, <720P, 30 fps, 40>}**

- **Often realized as *embedded bit streams***
  - **E.g., the stream with lower resolution is embedded in that with higher resolution**
  - **Provides successive refinement**

- **More details next week.**

Data Partitioning



- **Some parts of data are more important than others**
  - **Header > MV > Texture (Transform residual)**
- **Group important info and apply UEP (Unequal Error Protection)**

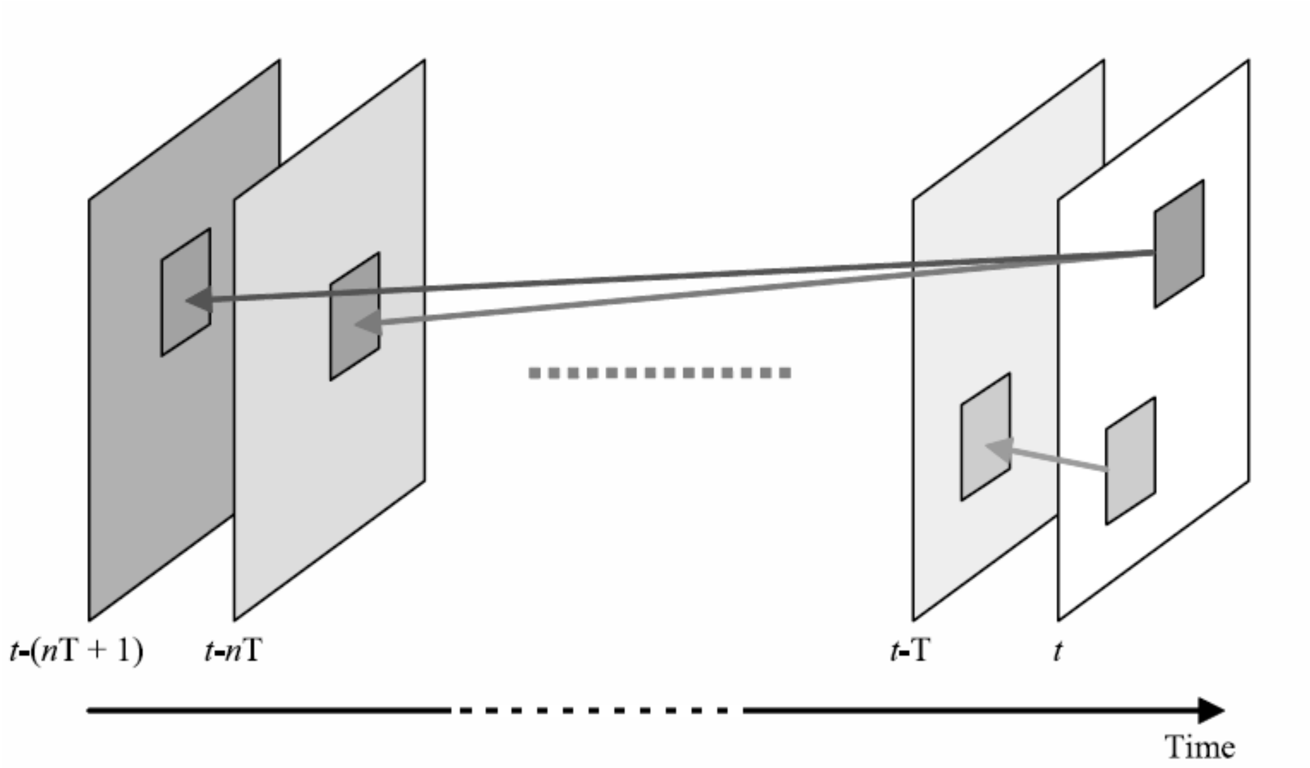
Redundant Slices

- H.264/AVC encoder can transmit redundant slices
  - Using possible different coding parameters
- H.264/AVC decoder uses them in case of errors
  - discards redundant slices if no errors
- Example:
  - Encoder can send a coarsely quantized (i.e., smaller size) version of ROI (e.g., face) as redundant slice
  - In case of loss, the decoder can display ROI, with lower quality

Flexible Reference Frame

- **MPEG-2 and H.263:**
  - 1 reference frame for P-frame and up to 2 for B-frame
- **But there could be similarity with more than one frame**
  - Target frame may be closer to frame r_1, which is different from predetermined reference frame r_2
  - More flexible prediction structure → higher compression and better error resiliency
- **MPEG-4 and H.263+:**
  - Supports Reference Picture Selection (RPS) → temporal prediction is possible from other correctly received frames
- **H.264/AVC:**
  - Generalized this to macroblock level

Flexible Reference Frame (cont'd)



- A MB can be predicted from multiple reference MBs or a combination of them in H.264/AVC
- Multiple frames are kept in the buffer for prediction

Summary

- Video coding: Spatial and temporal compression
- Most common (MPEG, H.26x)
  - Transform coding
  - Predictive coding
  - Hybrid coding
- Main steps for video coder and decoder
- Errors/losses can have severe impacts on video quality
  - Dependency among frames → error propagation
  - Tools for error resilience coding: Slice, FMO, partitioning, …
  - Error concealment (later)

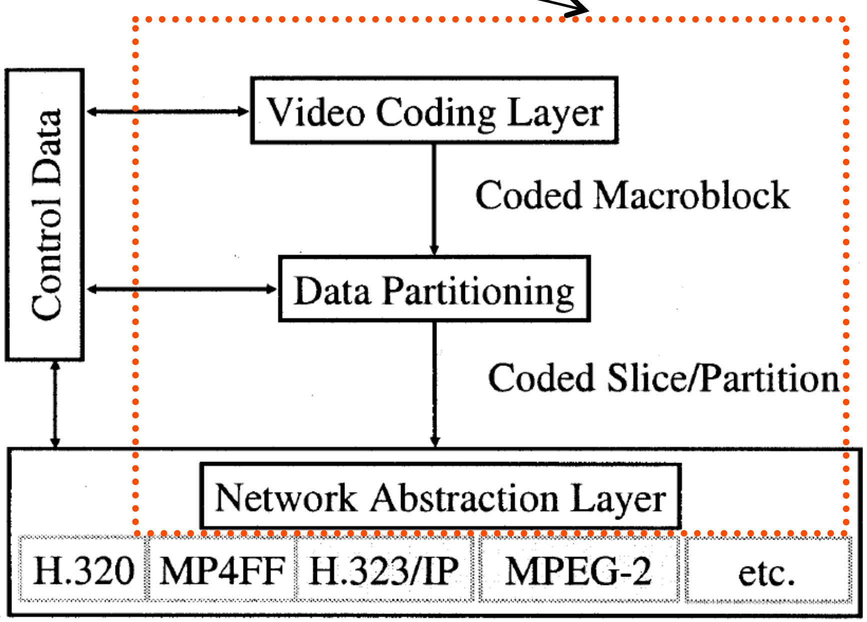
A Quick Overview of H.264/AVC

Design Goals of H.264/AVC

- **Cut the bit rate by at least half, compared to previous standards (MPEG-2)**
  - Reduce the network load or
  - Increase video quality
- **Make it flexible and suitable to various applications**
  - Replace MPEG-2 for pre-recorded videos
  - Replace H.263 for live videos
- **Keep the complexity manageable**
  - So it will be implemented at reasonable cost

Layered Design

- **Network Abstraction Layer (NAL)**
  - Formats video and meta data for variety of networks
  - Transport specific features
- **Video Coding Layer (VCL)**
  - Represents (encodes) video in an efficient way
  - Coding specific features



Network Abstraction Layer

- To be network friendly, supports various *transports*
  - RTP/IP for Internet applications
  - MPEG-2 streams for broadcast services
  - ISO File formats for storage applications
- Essentially are packets consist of video data
  - short packet header: one byte
- Support two types of transports
  - stream-oriented: no free unit boundaries ← use a 3-byte start code prefix
  - packet-oriented: start code prefix is a waste
- Have two types:
  - VCL units: data for video pictures
  - Non-VCL units: meta data and additional info

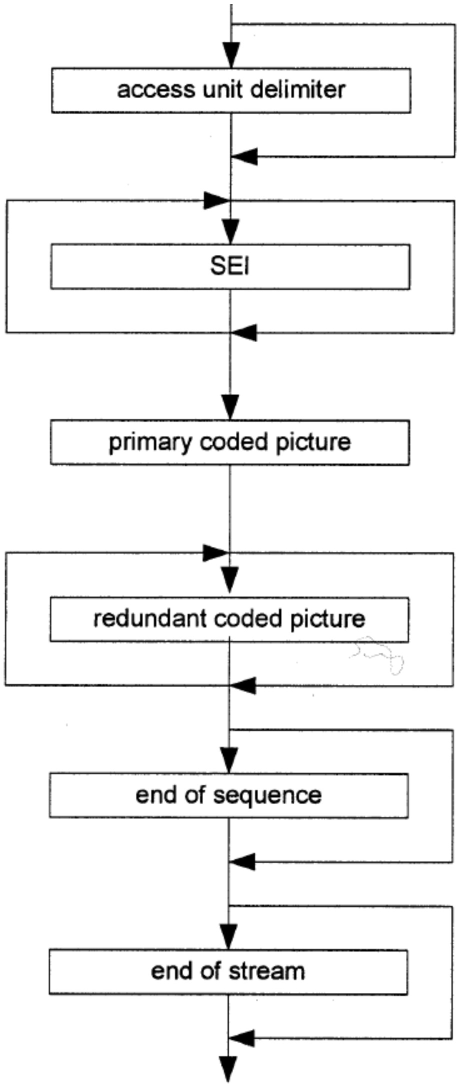
Network Abstract Layer Format

- **Each NAL unit contains an RBSP**
- **Raw Byte Sequence Payload (RBSP)**
  - **a set of data corresponding to coded video data or header information.**

Sequence of NAL units (Access Unit)

Access Units

- A set of NAL units
- Decoding an access unit results in one picture
- Structure:
  - Delimiter: for seeking in a stream
  - SEI (Supp. Enhancement Info.): timing and other info
  - primary coded picture: VCL
  - redundant coded picture: for error recovery



RBSP Format

Sequence parameter set	SEI	Picture parameter set	I slice	Picture delimiter	P slice	P slice	.....

RBSP type	Description
Parameter Set	'Global' parameters for a sequence such as picture dimensions, video format, macroblock allocation map (see Section 6.4.3).
Supplemental Enhancement Information	Side messages that are not essential for correct decoding of the video sequence.
Picture Delimiter	Boundary between video pictures (optional). If not present, the decoder infers the boundary based on the frame number contained within each slice header.
Coded slice	Header and data for a slice; this RBSP unit contains actual coded video data.
Data Partition A, B or C	Three units containing Data Partitioned slice layer data (useful for error resilient decoding). Partition A contains header data for all MBs in the slice, Partition B contains intra coded data and partition C contains inter coded data.
End of sequence	Indicates that the next picture (in decoding order) is an IDR picture (see Section 6.4.2). (Not essential for correct decoding of the sequence).
End of stream	Indicates that there are no further pictures in the bitstream. (Not essential for correct decoding of the sequence).
Filler data	Contains 'dummy' data (may be used to increase the number of bytes in the sequence). (Not essential for correct decoding of the sequence).

Video Sequences and IDR Frames

- **Sequence: a sequence of decodable NAL units**
  - With one sequence parameter set
  - Starts with an *instantaneous decoding refresh* (IDR) access unit
- **IDR frames: random access points**
  - Intra-coded frames
  - No future frame will refer to frames prior to an IDR frame ← main difference from I-frames
  - Decoders flush buffered reference pictures once seeing an IDR frame

Video Coding Layer Features (1/3)

- Features for enhancement of prediction
  - Directional spatial prediction for intra coding
  - Variable block-size motion compensation with small block size
  - Quarter-sample-accurate motion compensation
  - Motion vectors over picture boundaries
  - Multiple reference picture motion compensation
  - Decoupling of referencing order form display order
  - Decoupling of picture representation methods from picture referencing capability
  - Weighted prediction
  - Improved “skipped” and “direct” motion inference
  - In-the-loop deblocking filtering

Video Coding Layer Features (2/3)

- **Features for improved coding efficiency**
  - Small block-size transform
  - Exact-match inverse transform
  - Short word-length transform
  - Hierarchical block transform
  - Arithmetic entropy coding
  - Context-adaptive entropy coding

Video Coding Layer Features (3/3)

- **Features for robustness to data errors/losses**
  - Flexible slice size
  - Flexible macroblock ordering (FMO)
  - Arbitrary slice ordering (ASO)
  - Redundant pictures
  - Data Partitioning
  - SP/SI synchronization/switching pictures

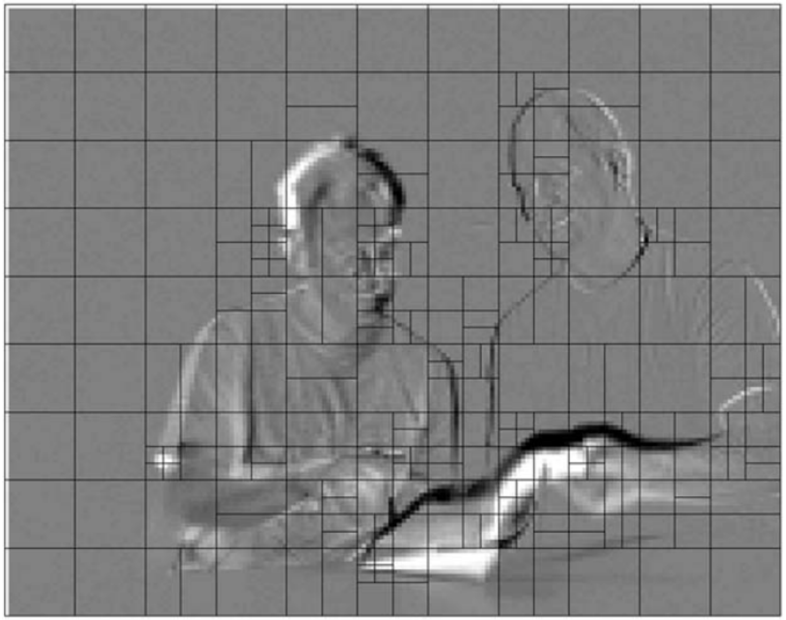
H.264 Slice Modes

Recall: Slice is a set of MBs that can be decoded

Slice Type	Description	Profile(s)
I (Intra)	Contains only I macroblocks (each block or MB is predicted from previously coded data within the same slice).	All
P (Predicted)	Contains P macroblocks (each MB is predicted from one list 0 reference picture) and/or I MBs.	All
B (Bi-predictive)	Contains B macroblocks (each MB is predicted from a list 0 and/or a list 1 reference picture) and/or I macroblocks.	Extended and Main
SP (Switching P)	Facilitates switching between coded streams; contains P and/or I macroblocks.	Extended
SI (Switching I)	Facilitates switching between coded streams; contains SI macroblocks (a special type of intra coded MB).	Extended

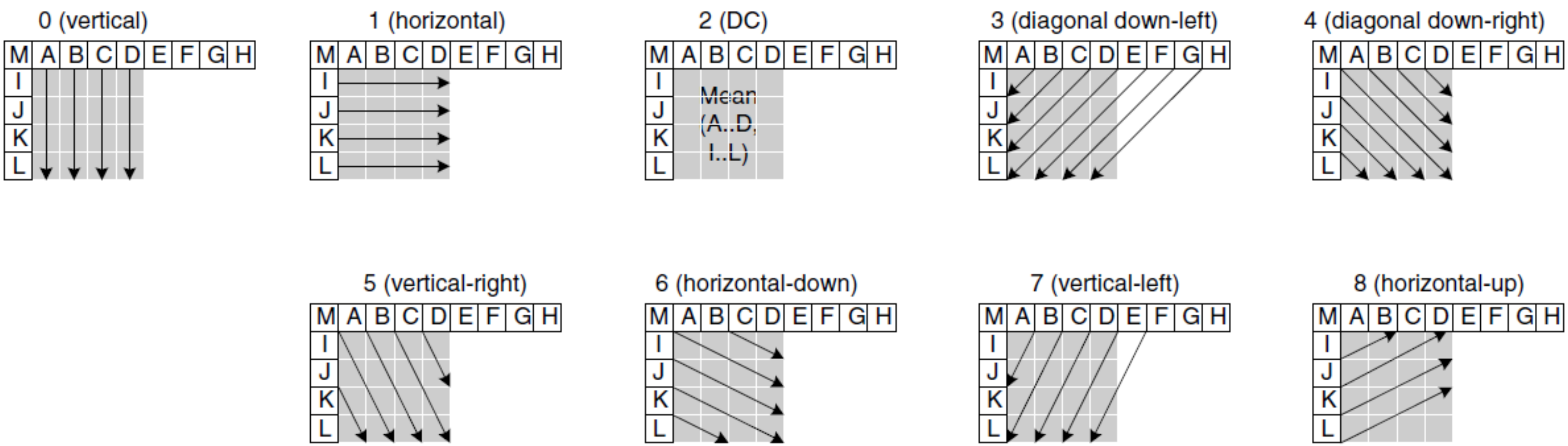
Inter Prediction

- **H.264/AVC supports a range of block sizes (from 16 × 16 down to 4×4) and fine *subsample* (1/4-pixel) motion vectors**
- **Partitioning MBs into motion compensated sub-blocks of varying size is known as tree structured motion compensation**
- **Intuitions:**
  - **MV are expensive**
  - **Smooth areas → larger**
  - **Detailed areas→ smaller**



Intra Prediction

- **Intra prediction in H.264 is conducted in the spatial domain**
- **Intra prediction is restricted to *Intra-coded* neighboring MBs ← to avoid error propagation**
- **4x4 (for detailed areas, 9 modes) and 16x16 (for smooth areas, 4 modes) predictions**



Deblocking Filter

- Smooth out block edges
- After inverse DCT (both encoder/decoder)
- Idea:
  - large diff. between pixels across block edges →blocking artifacts
  - But if the diff. is very large → probably real edges in original frame

Original Frame

Reconstructed, QP=36 (no filter)

Reconstructed, QP=36 (with filter)

4x4 Integer Transform

- **Why smaller transform**
  - Only use add and shift, an exact inverse transform is possible ← no decoding mismatch
  - Not too much residue to code ← ME is good enough
  - Less noise around edge (ringing or mosquito noise)
  - Less computations and shorter data type (16-bit)
- **An approximation to 4x4 DCT**

$$Y = C_f X C_f^{\mathrm{T}} \otimes E_f = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} & & & \\ & \mathbf{X} & & \\ & & & \\ & & & \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}$$

$$a = \frac{1}{2}, \qquad b = \sqrt{\frac{2}{5}}, \qquad d = \frac{1}{2}$$

Quantization

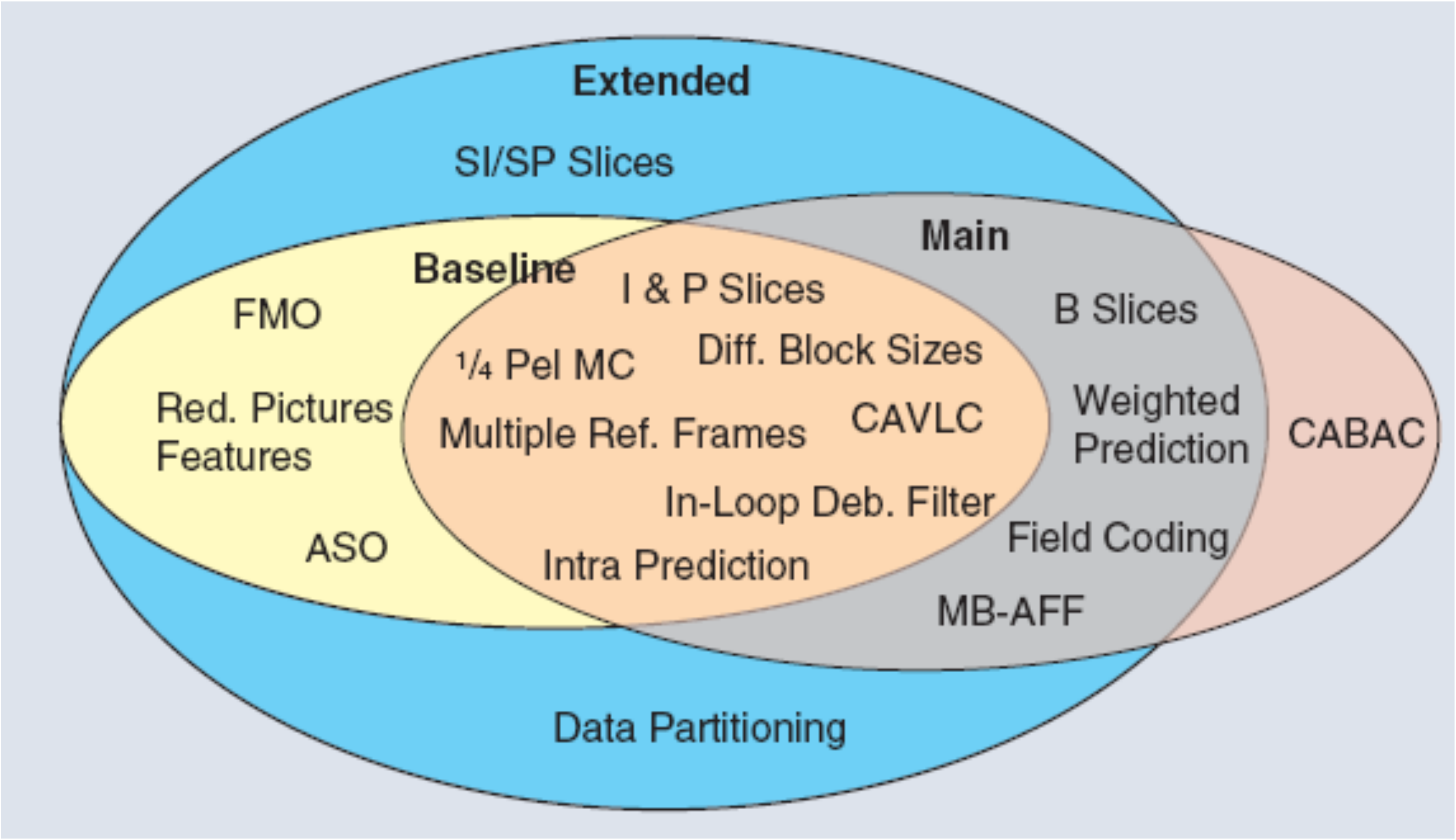
- A total of 52 values of Q_{step} are supported by the standard, indexed by a Quantisation Parameter, QP.
- Q_{step} doubles in size for every increment of 6 in QP.

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	…
QStep	0.625	0.6875	0.8125	0.875	1	1.125	1.25	1.375	1.625	1.75	2	2.25	2.5	…
QP	…	18	…	24	…	30	…	36	…	42	…	48	…	51
QStep		5		10		20		40		80		160		224

Entropy Coding

- Non-transform coefficients: Exp-Golomb ← loseless
- Transform coefficients:
  - Context-Adaptive Variable Length Coding (CAVLC)
    - several VLC tables are switched dep. on prior transmitted data ← better than a single VLC table
  - Context-Adaptive Binary Arithmetic Coding (CABAC)
    - flexible symbol probability than CAVLC ← 5 – 15% rate reduction
    - multiplication free

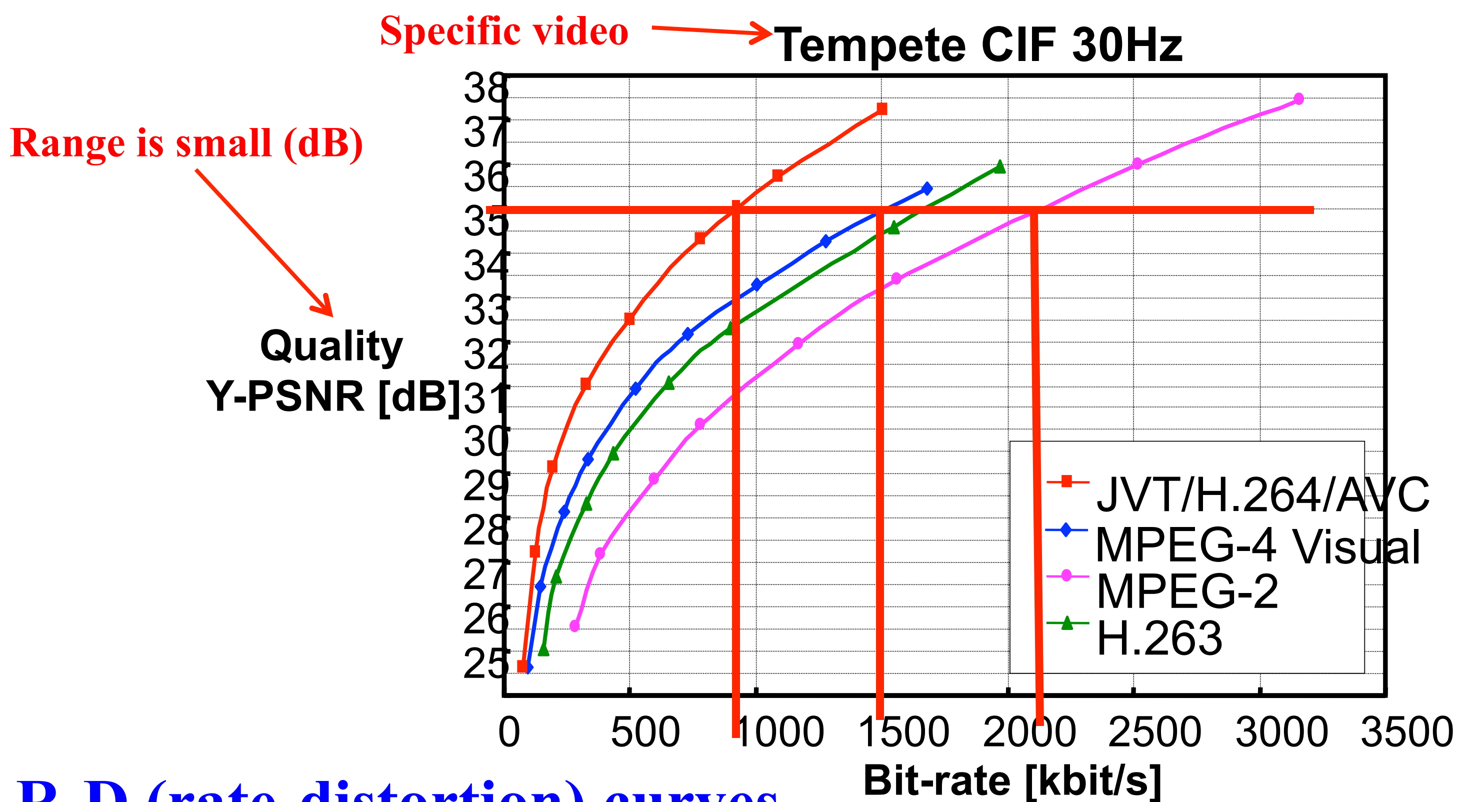
H.264 Profiles



Potential Applications

- **Baseline (low latency)**
  - H.320 conversational video services
  - 3GPP conversational H.324/M services
  - H.323 with IP/RTP
  - 3GPP using IP/RTP and SIP
  - 3GPP streaming using IP/RTP and RTSP
- **Main (moderate latency)**
  - Modified H.222.0/MPEG-2
  - Broadcast via satellite, cable, terrestrial or DSL
  - DVD and VOD
- **Extended**
  - Streaming over wired Internet
- **Any (no requirement on latency)**
  - 3GPP MMS
  - Video mail

Performance of H.264/AVC



- **R-D (rate-distortion) curves**
  - Encode at different bit rates
  - Decode and compare quality (distortion) relative to original video
- **EX: to get 35 dB ➔ ~ 900 Kbps (AVC), 2.1 Mbp (MPEG-2)**

Conclusion

- H.264 includes many coding tools for high coding efficiency
- H.264 supports various kinds of multimedia communication applications
- H.264 indeed achieves the design goal of reducing the bit rate by half