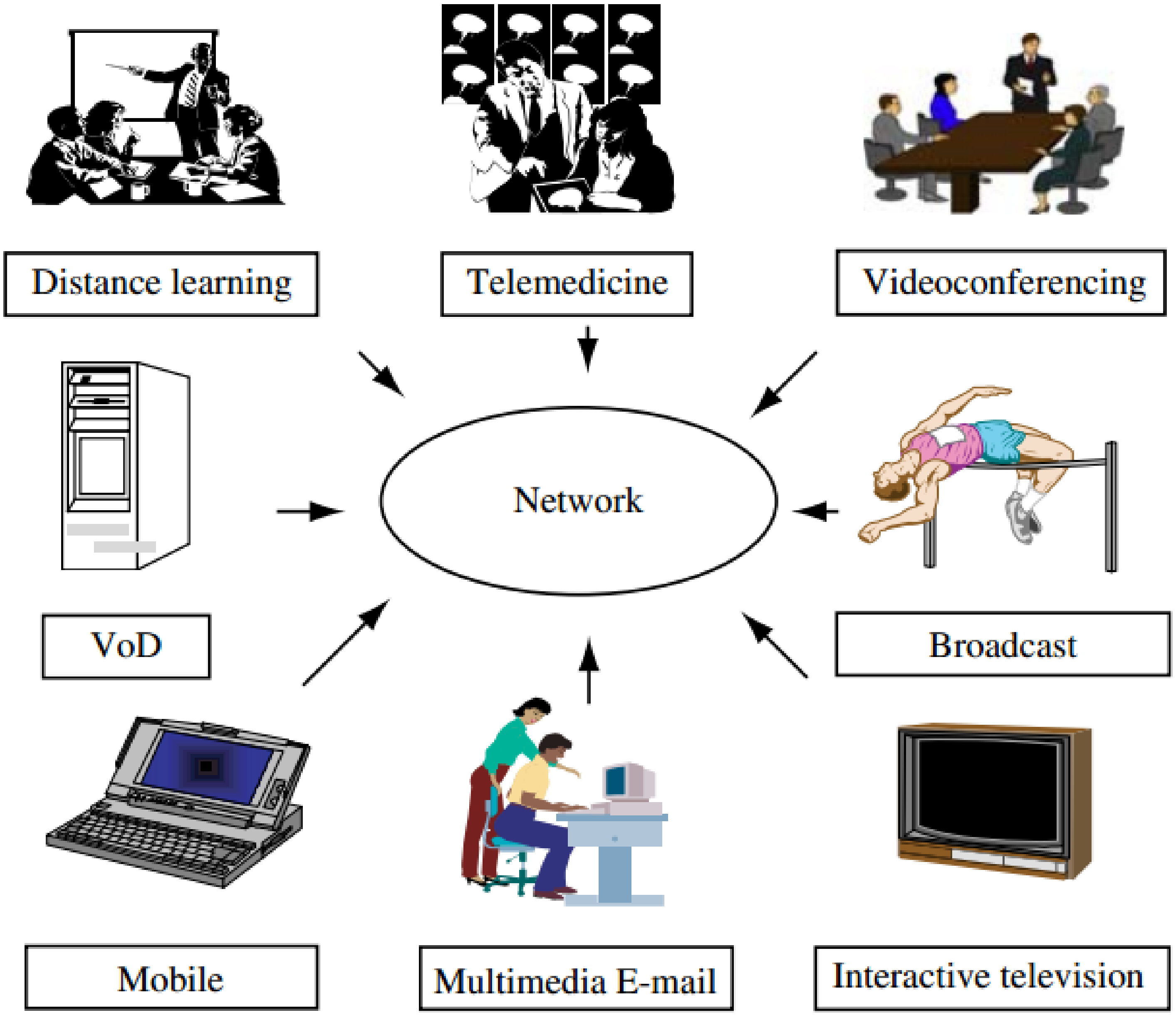
Video Compression Techniques

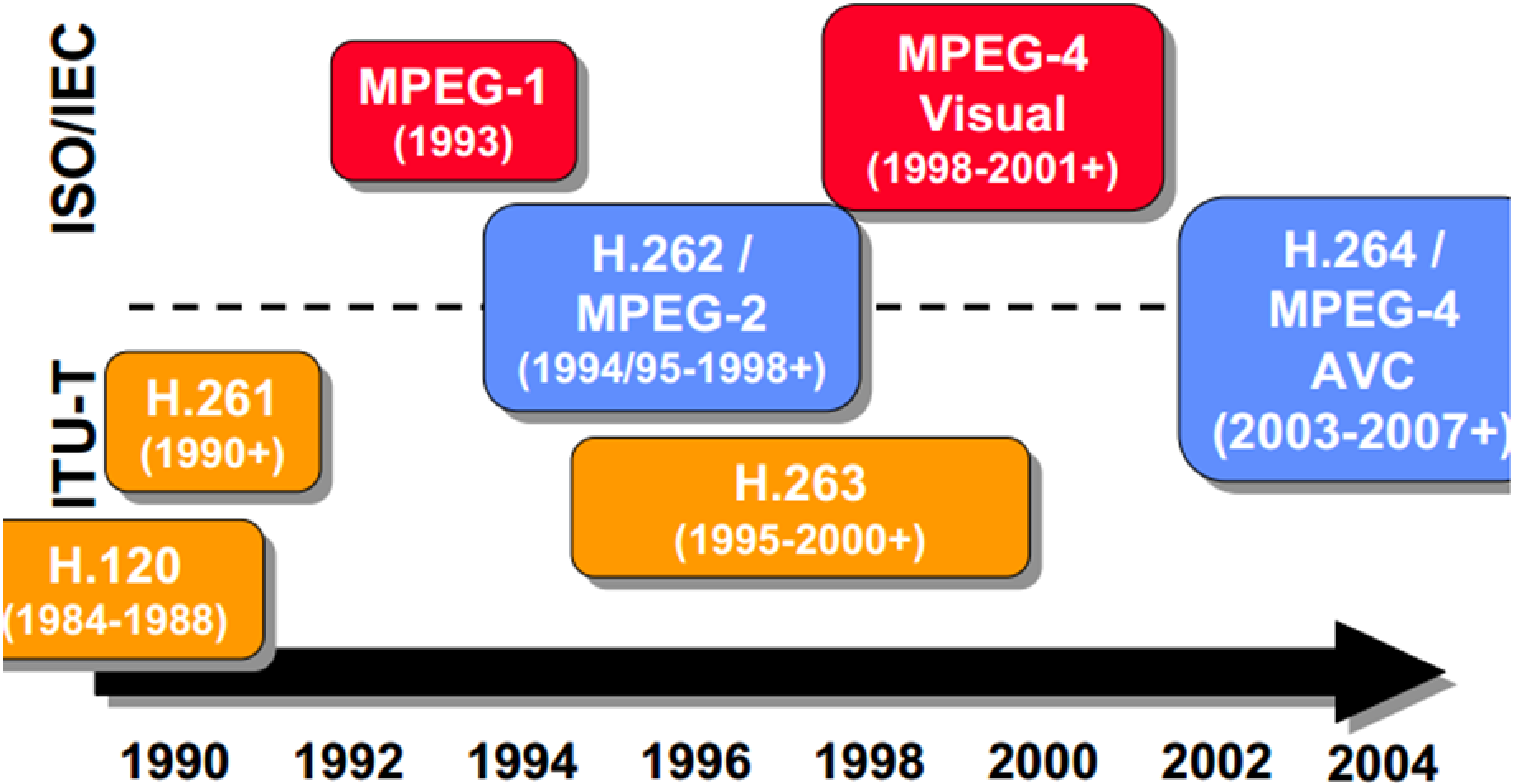
Santosh Chapaneri,
Assistant Professor,
EXTC, SFIT

Why Video Compression?



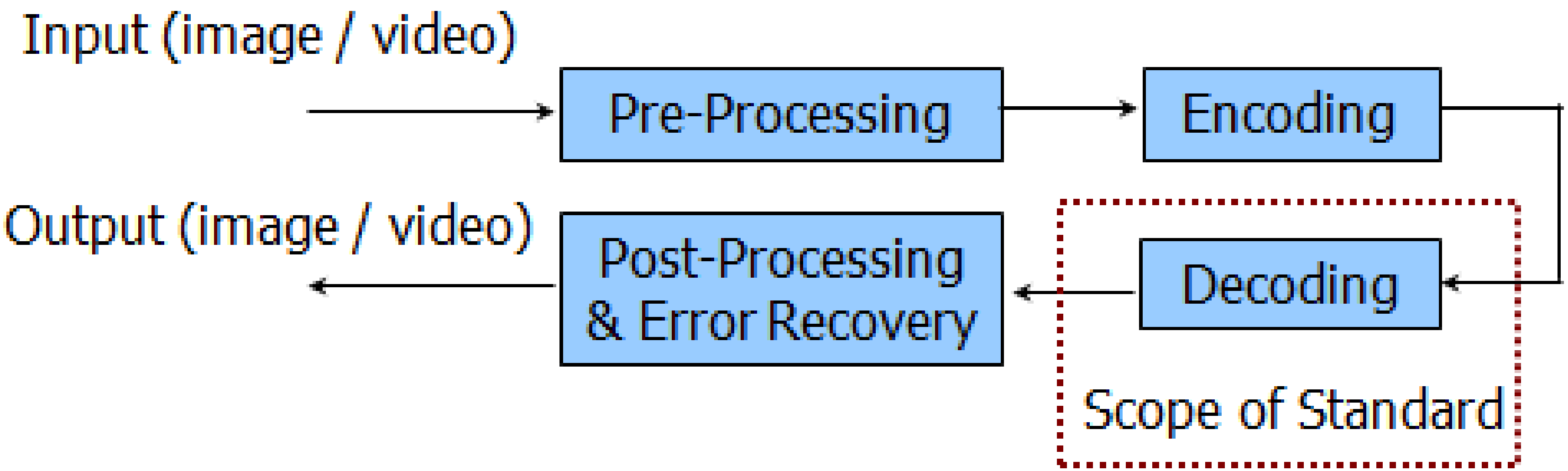
Embedded Video Codecs

Chronology of Video Standards

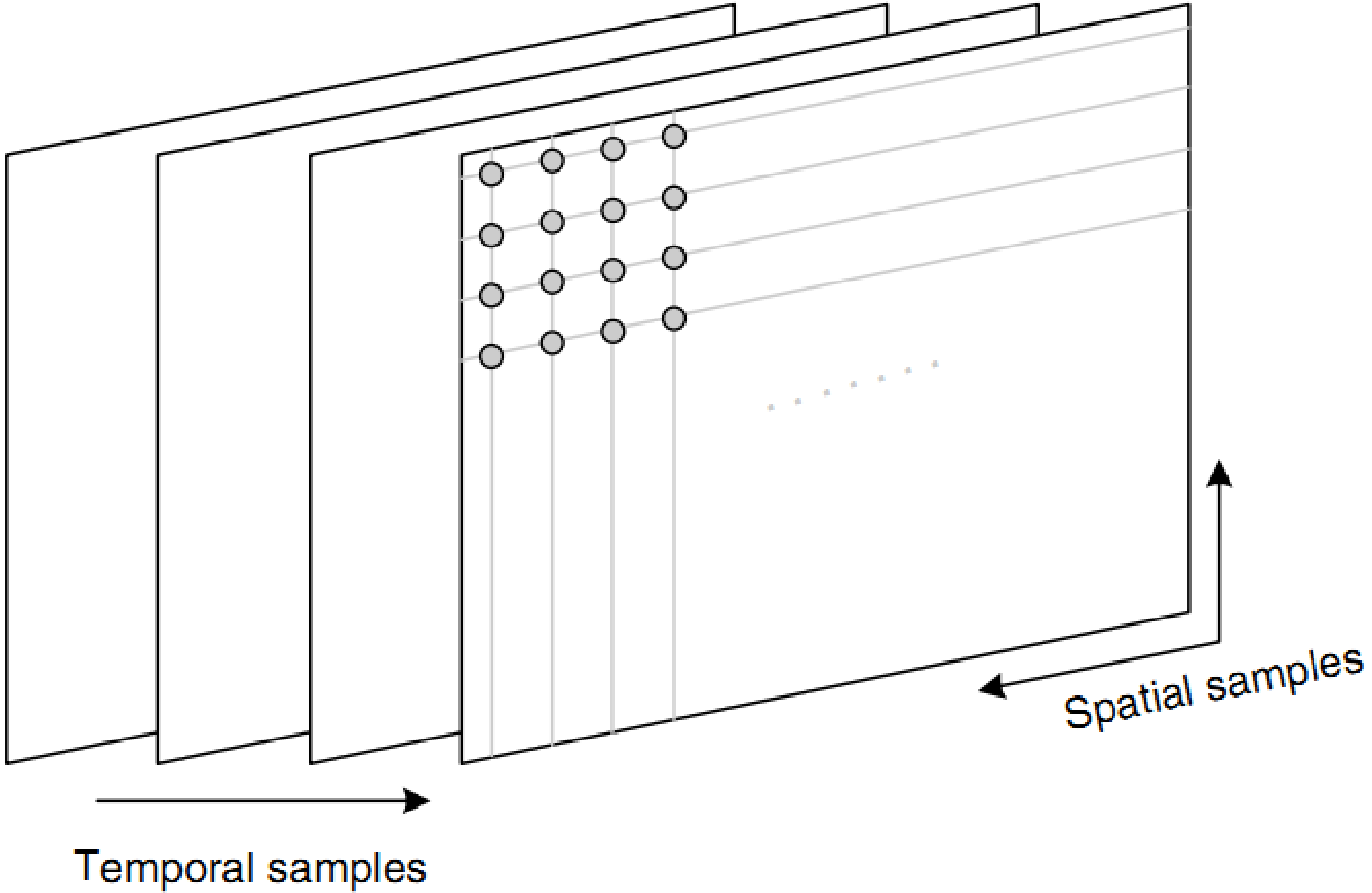


Scope of Video Standards

- Only the Syntax and Decoder are standardized:
  - Optimization beyond the obvious
  - Complexity reduction for implementation
  - Provides no guarantees of quality



Video Sequence



Video Resolutions

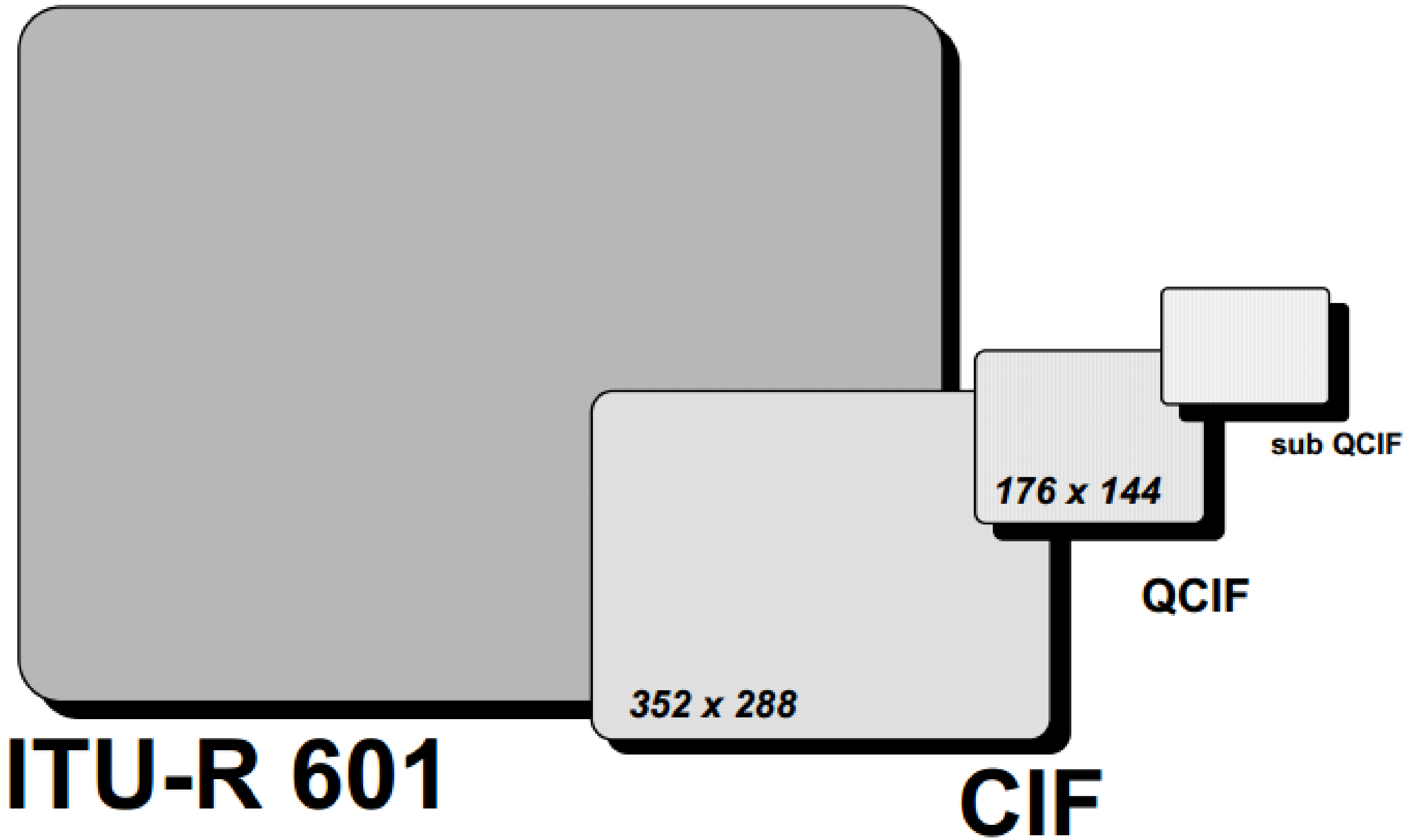
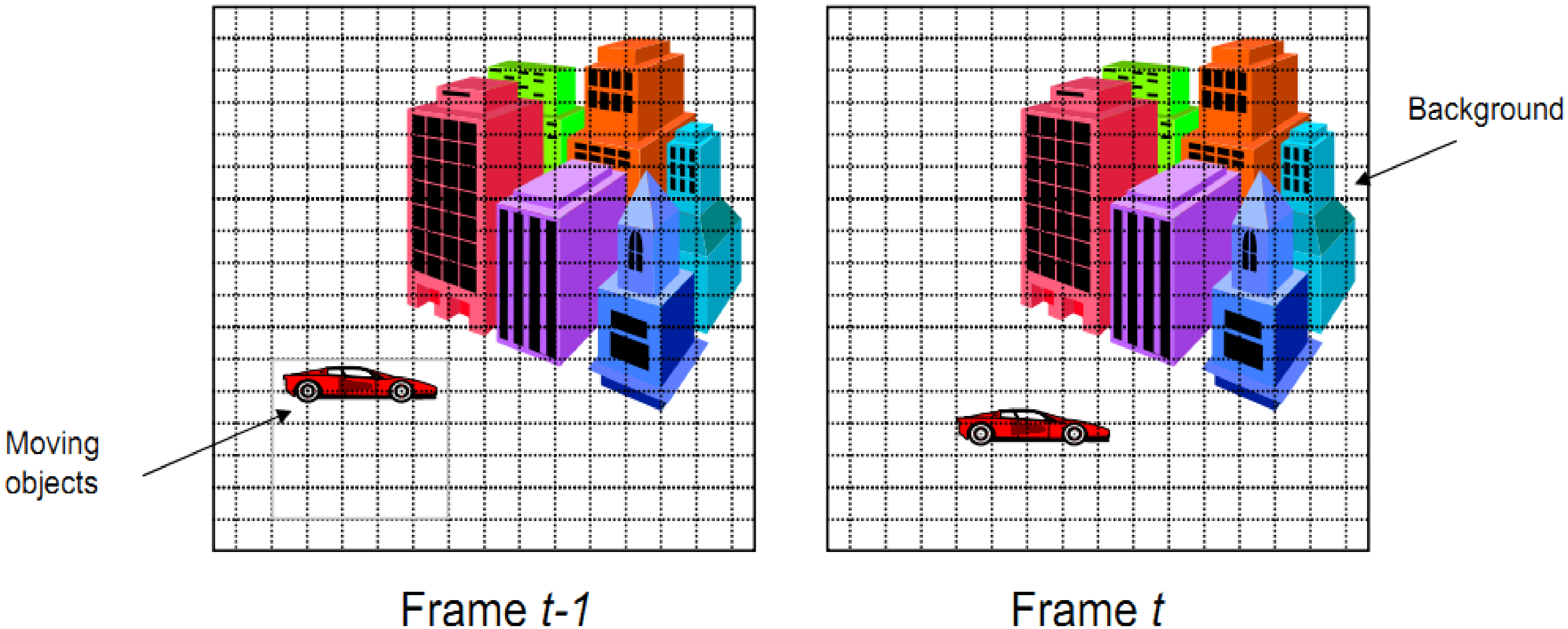


Image Compression Overview

- Why can we compress an image
  - Adjacent pixels are correlated (have similar color values)
- How to compress (the JPEG way)
  - Use transform to decorrelate the signal (DCT)
  - Quantize the DCT coefficients
  - Runlength code the quantized indices
    - Zigzag ordering
    - Huffman coding each pair (zero runlength, non-zero value)
- What is different with video?
  - We can apply JPEG to each video frame (Motion-JPEG)
  - But we can do more than that to achieve higher compression!

Typical Video Sequence



Adjacent frames are similar and changes are due to object or camera motion

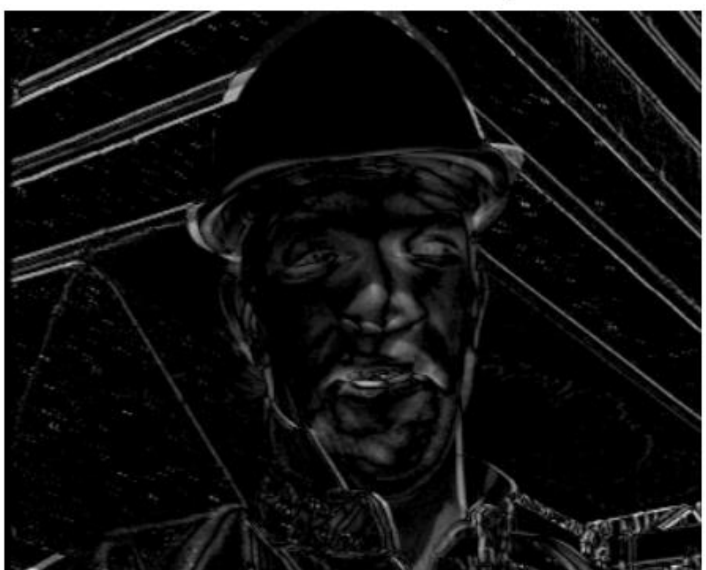
--- Temporal correlation

Temporal Redundancy

Frame 66

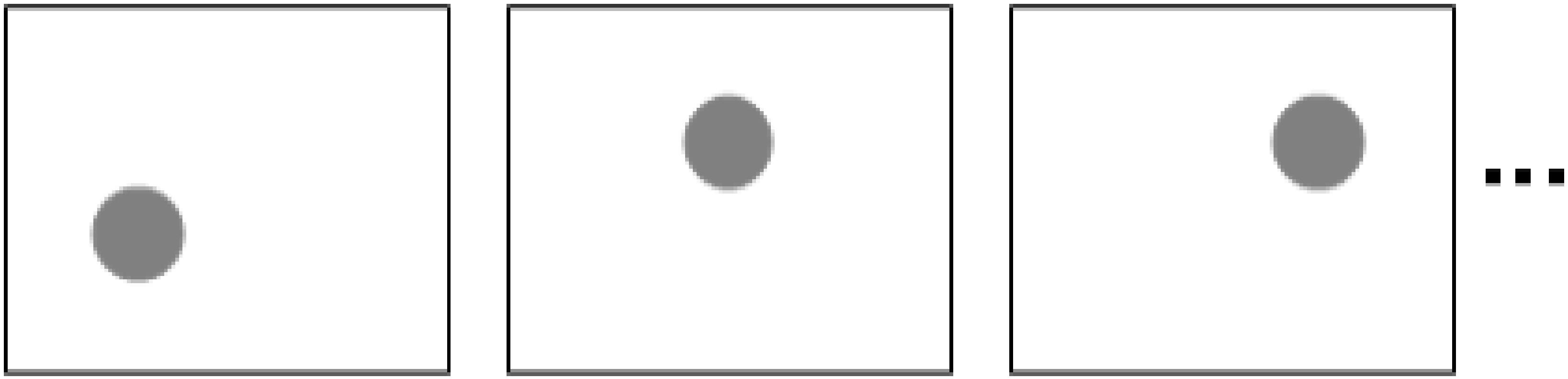
Frame 69

Absolute Difference w/o Motion Compensation

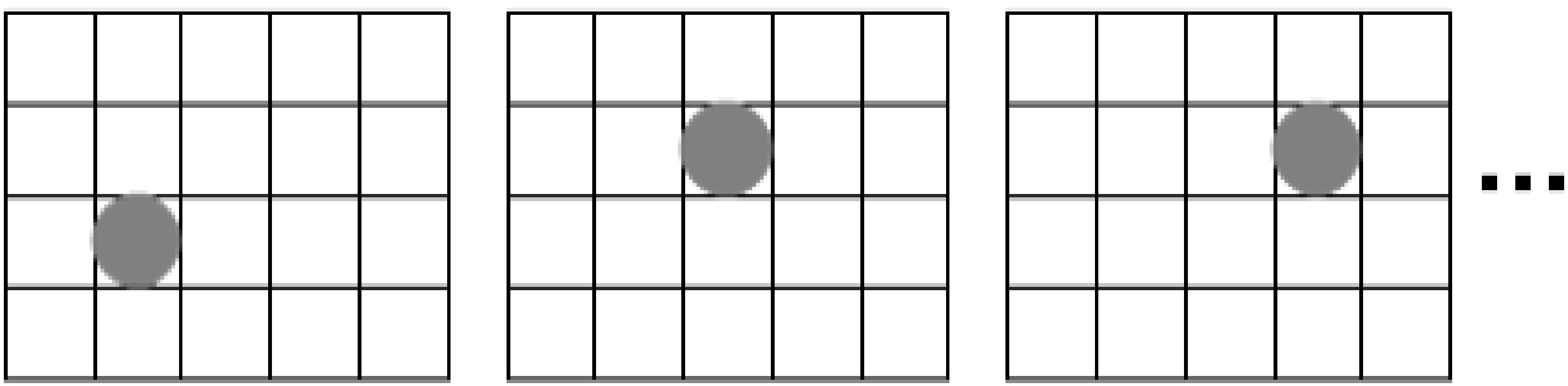


Absolute Difference with Motion Compensation

Motion Estimation

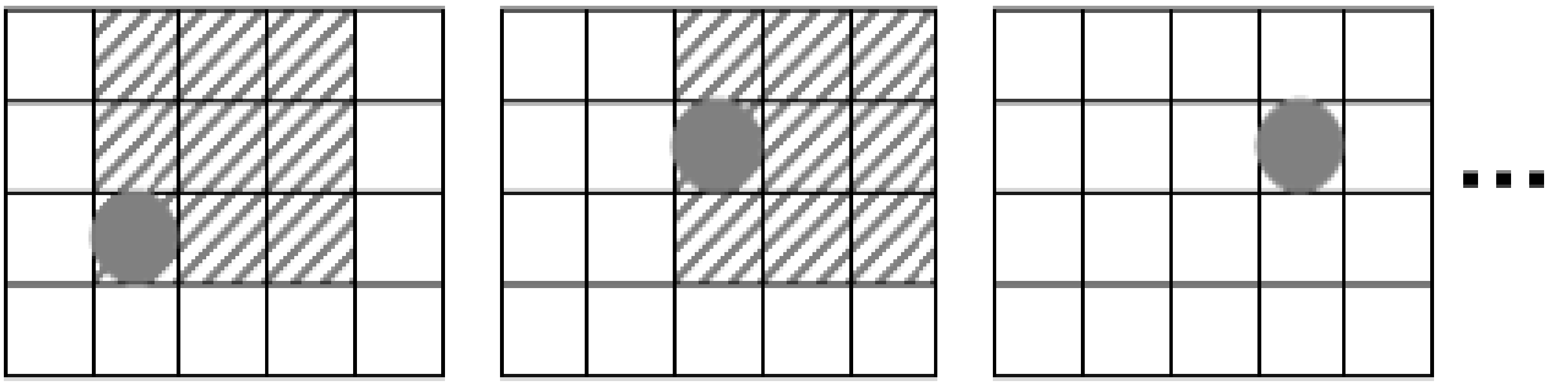


Video of a 'ball' moving up and to the right across three consecutive frames

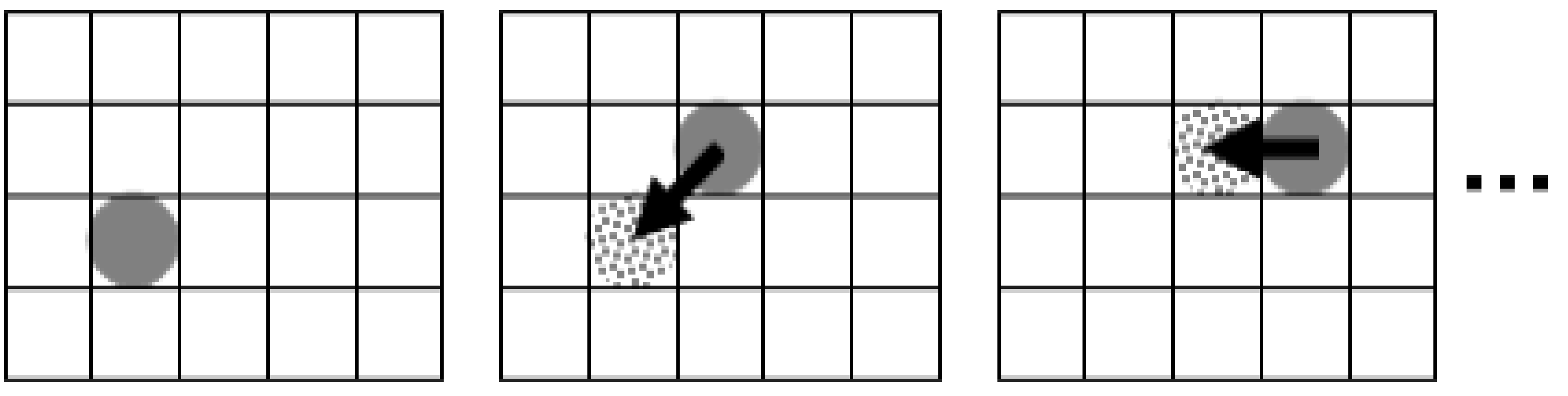


The frames are divided into macroblocks

Motion Estimation

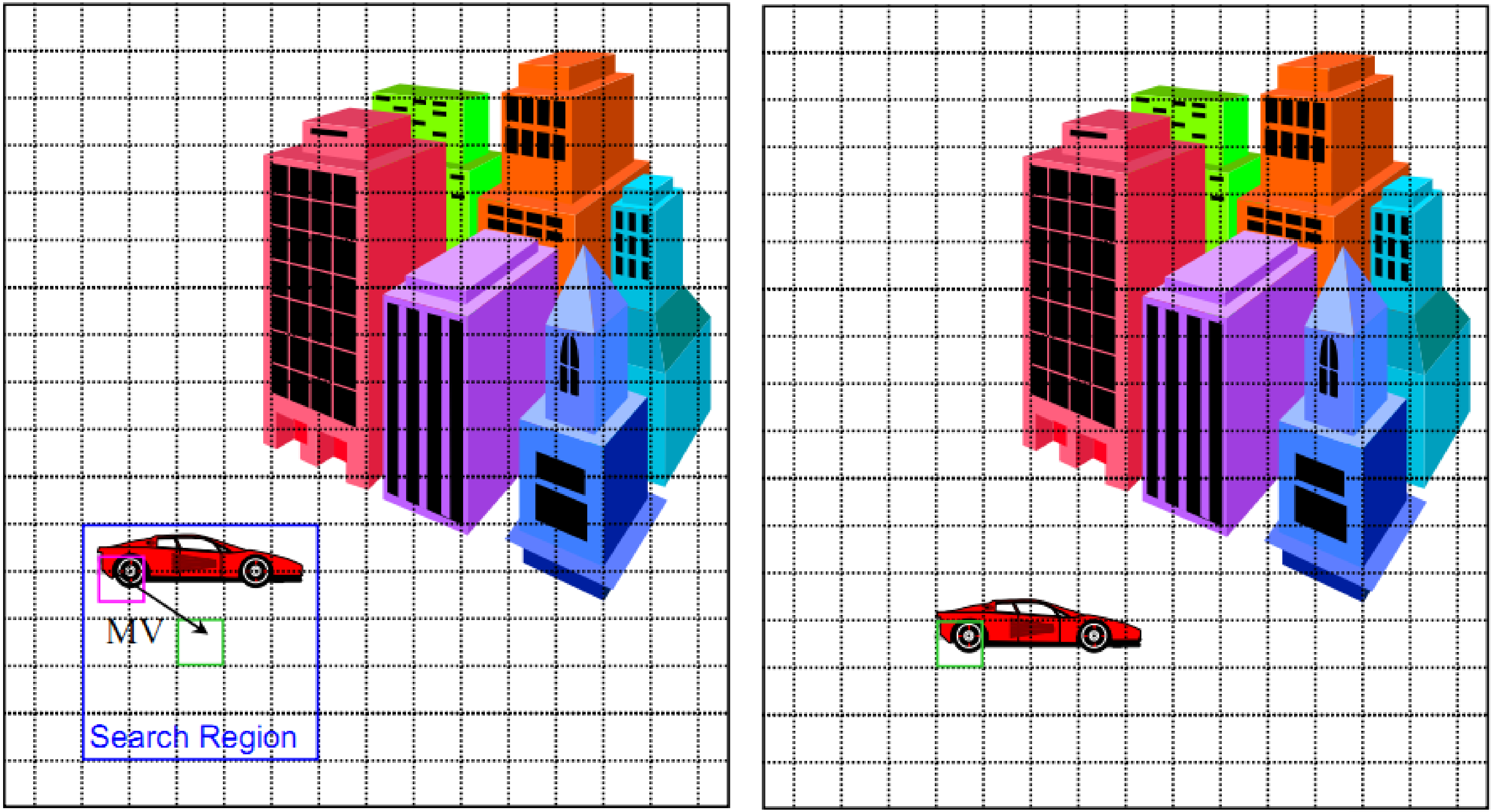


ME searches in the shaded areas for where the ball macroblocks moved from



ME delivers motion vectors that point to the previous macroblocks to re-use

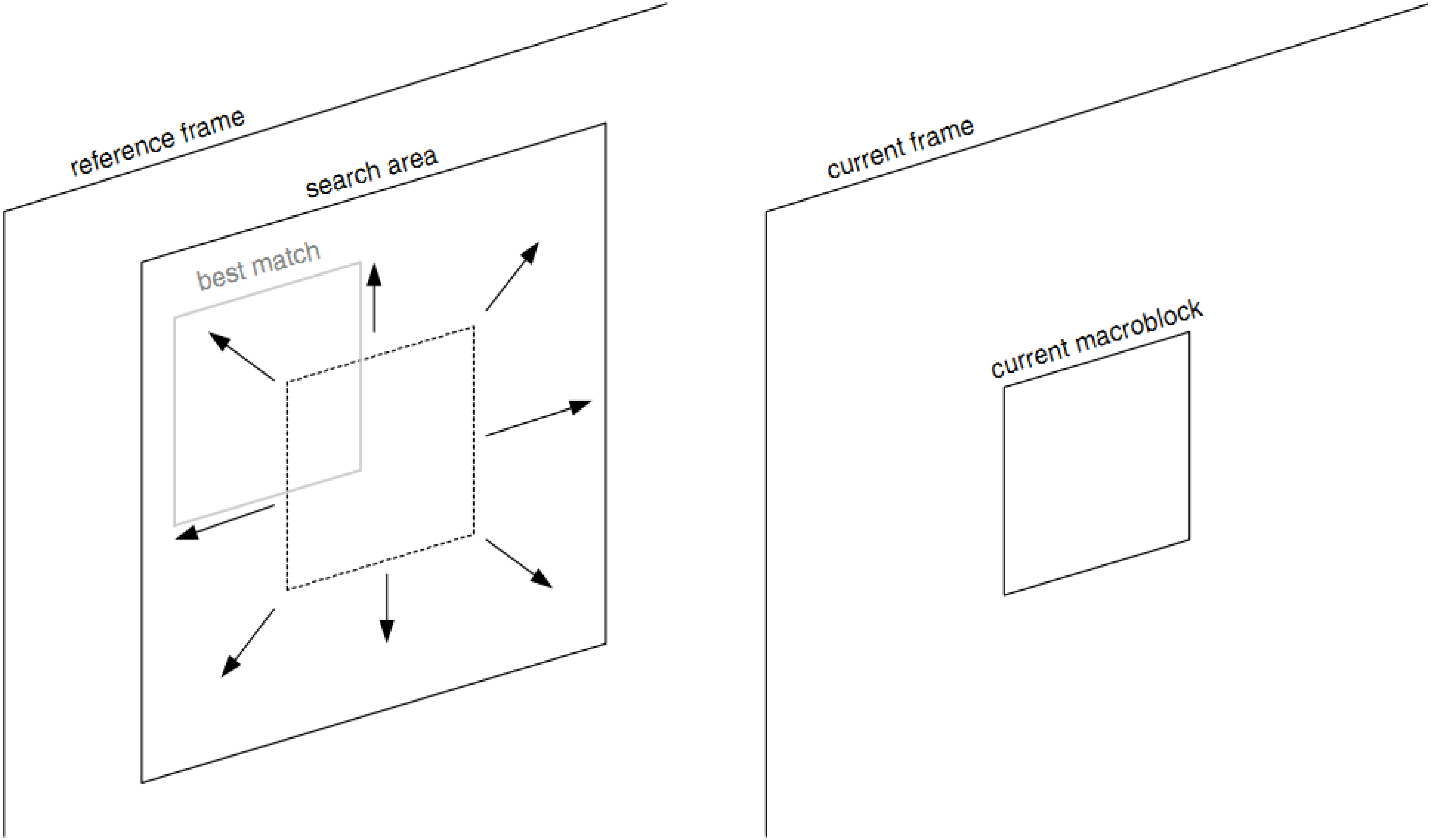
Motion Estimation



Frame *t-1*
(Reference Frame)

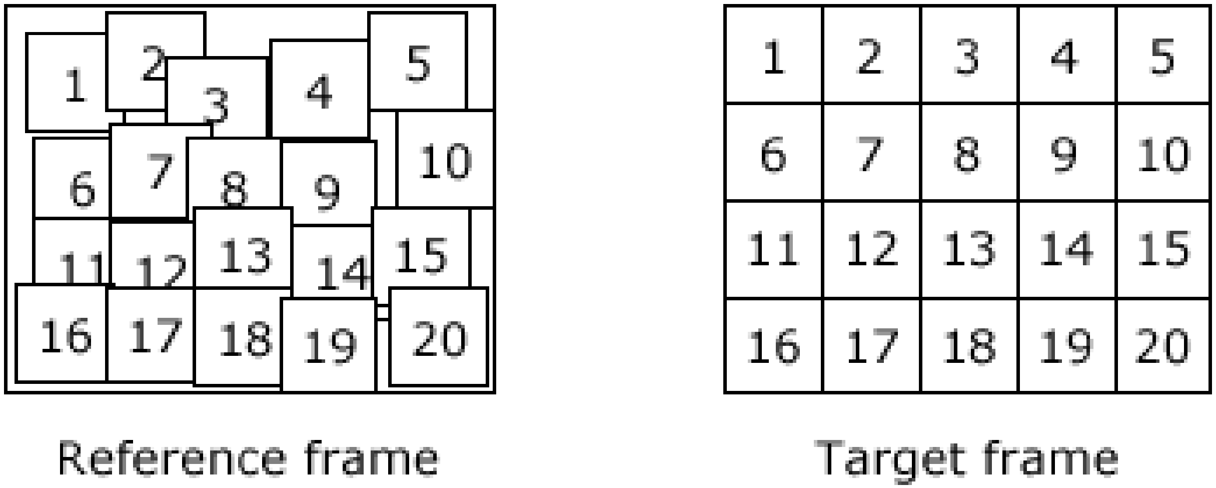
Frame *t*
(Predicted frame)

Motion Estimation



Motion Estimation

- Find "best" matching blocks in previous frames (motion vectors), then apply the same principles of coding as for JPEG

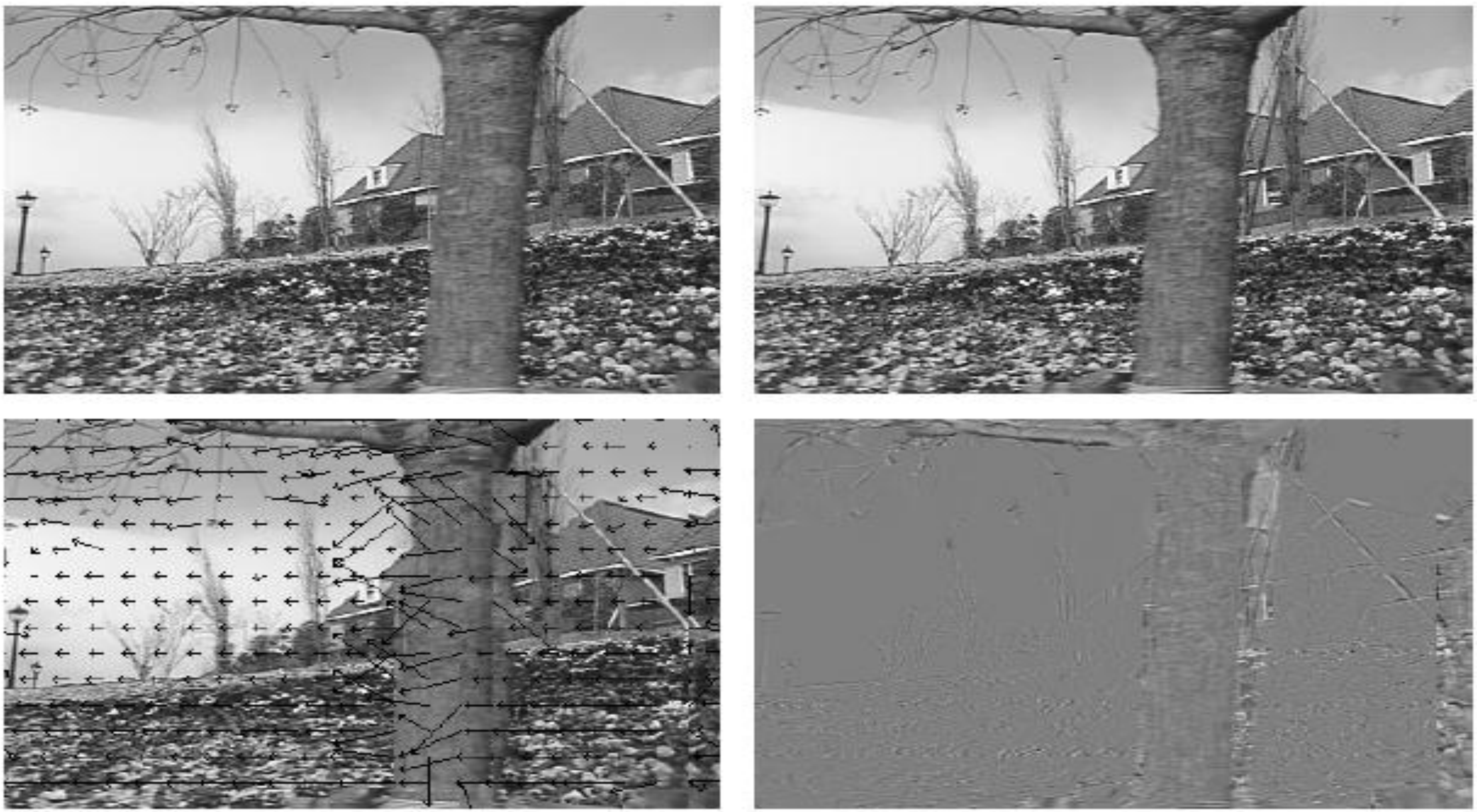


- Full search, Logarithmic, Diamond, Pyramidal search
- **Full search** will give accurate answer, searches for all possible displacements; but **most time consuming**
- New standards for video compression also supports half-pixel and quarter-pixel motion estimation
- Find motion vectors in estimation process and compensate by prediction from the displacement

Motion Compensation

Previous frame

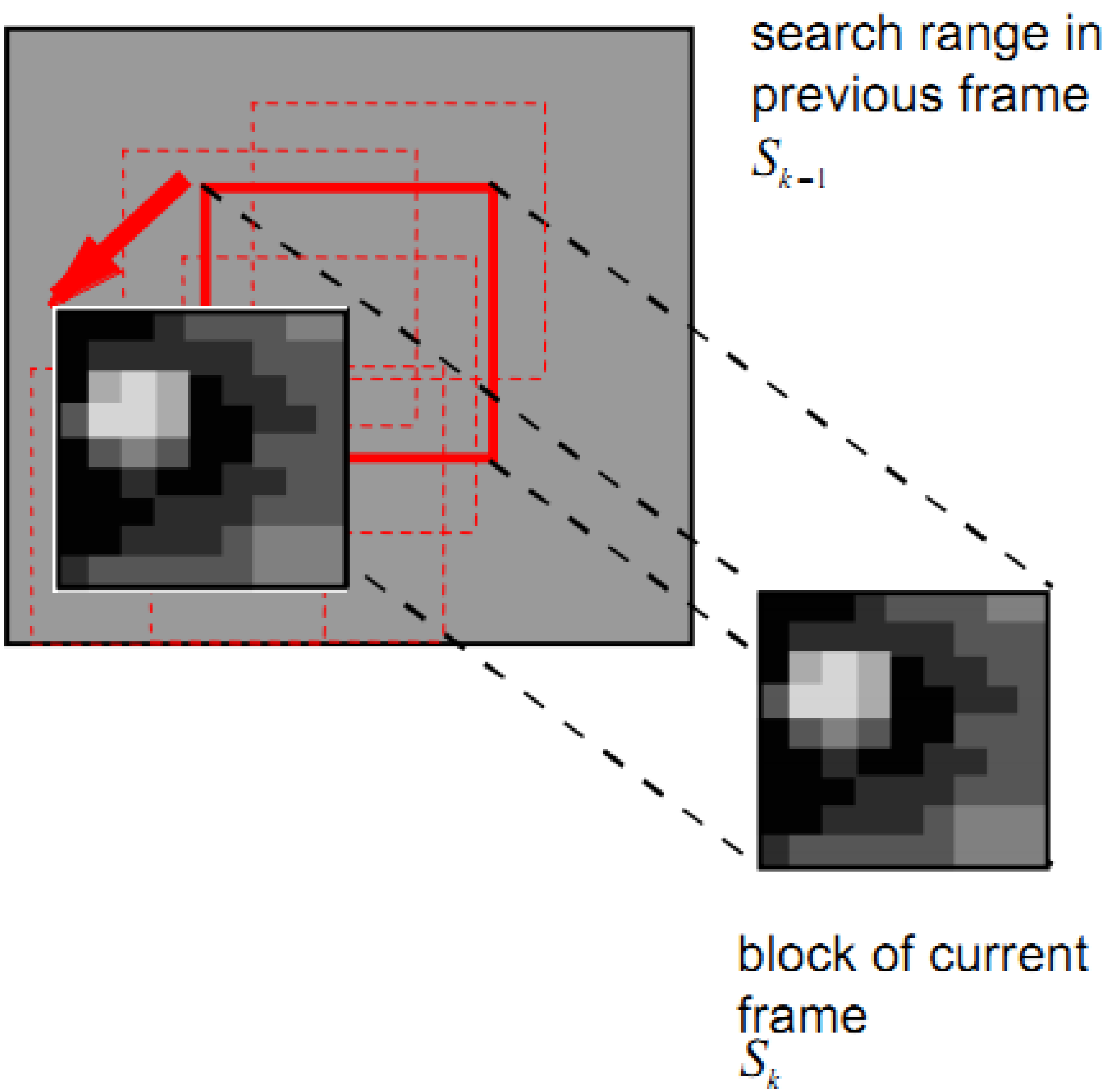
Current frame



Current frame with displacement vectors

Motion-compensated Prediction error

Block Matching Algorithm (BMA)



- Subdivide every image into square blocks.
- Find one displacement vector for each block.
- Within a search range, find a best „match“ that minimizes an error measure.
- Intelligent search strategies can reduce computation.

BMA Matching Criteria

- *Sum of Squared Differences* to determine similarity

Sum all values in measurement window

Current image

Previous image

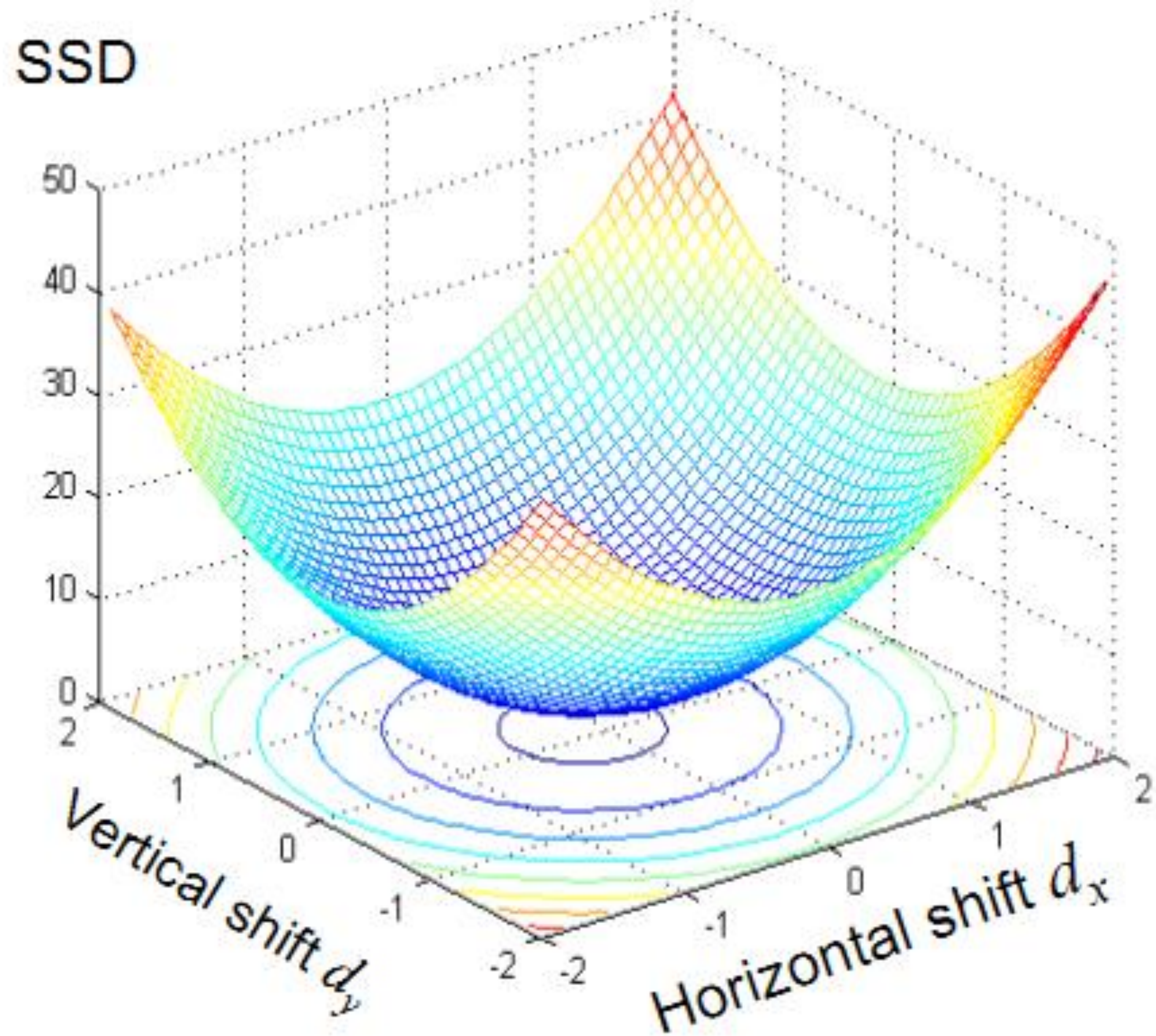
$$SSD(d_x, d_y) = \sum_{\text{msmnt window}} \left[S_k(x, y) - S_{k-1}(x + d_x, y + d_y) \right]^2$$

Horizontal shift

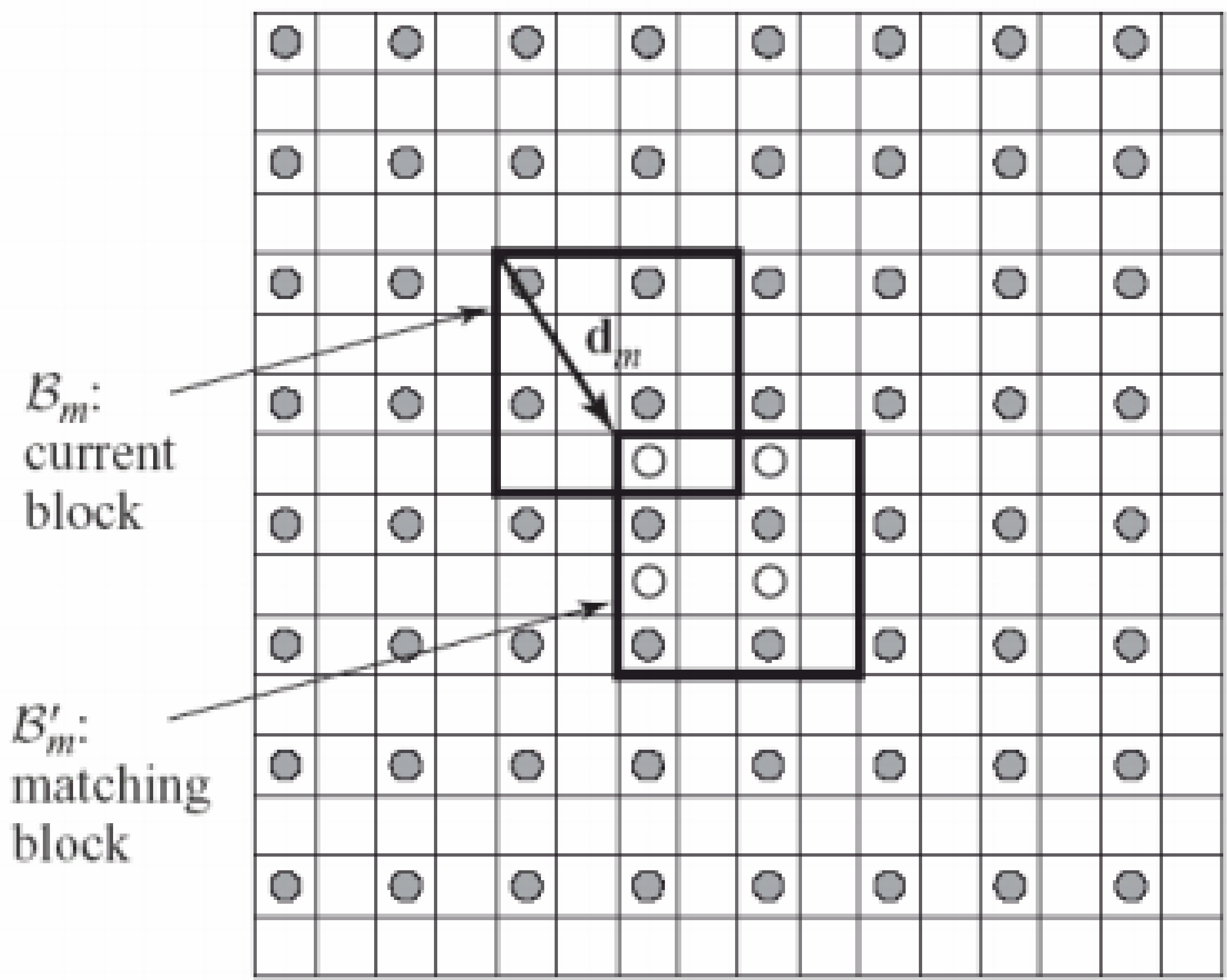
Vertical shift

- Alternative matching criteria: SAD (*Sum of Absolute Differences*), cross correlation, . . .
- Only integer pixel shifts are possible

Minimum SSD for a MB

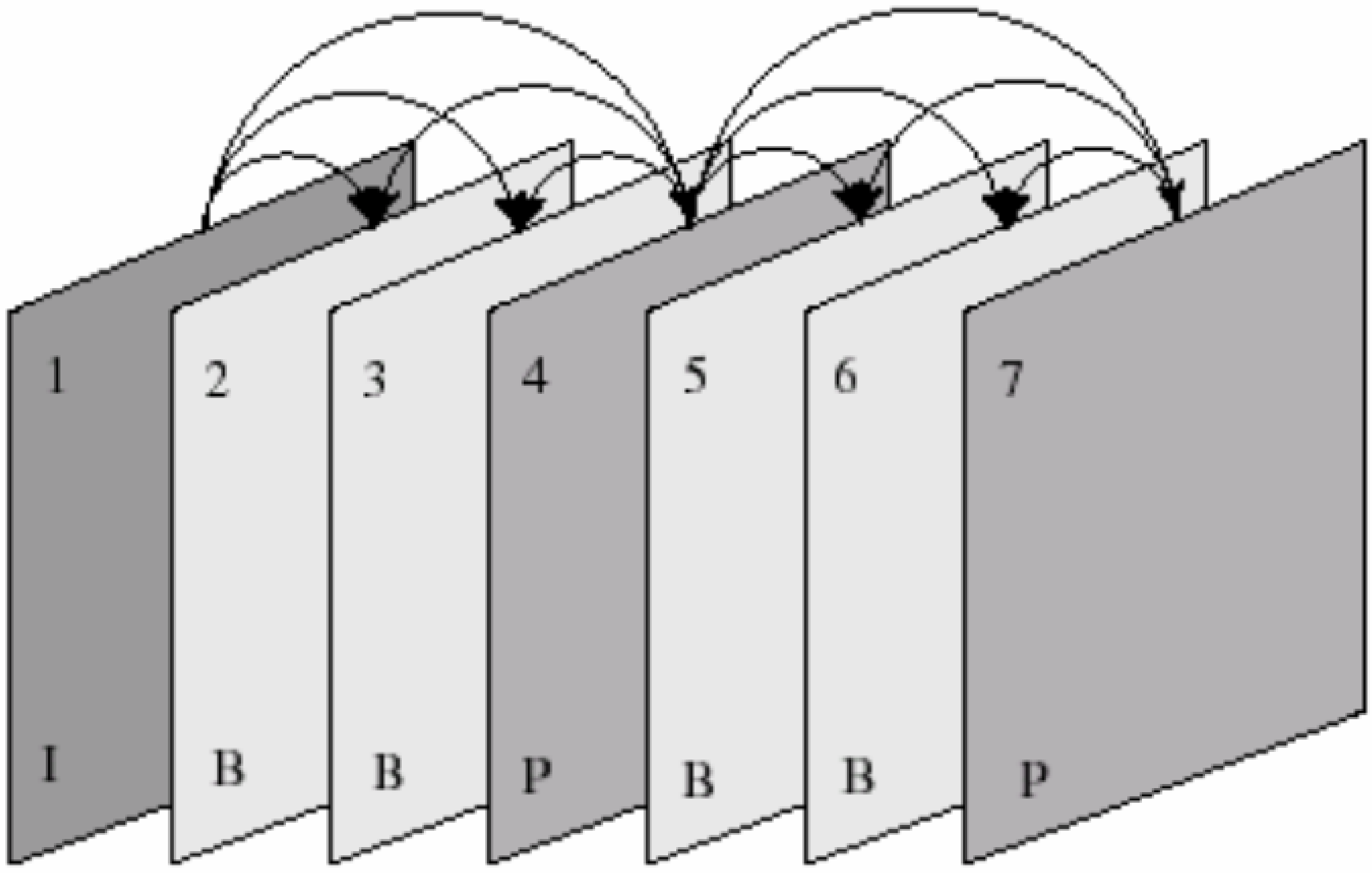


Half-Pel BMA



$\mathbf{d}_m$
$\mathcal{B}_m$: current block
$\mathcal{B}'_m$: matching block

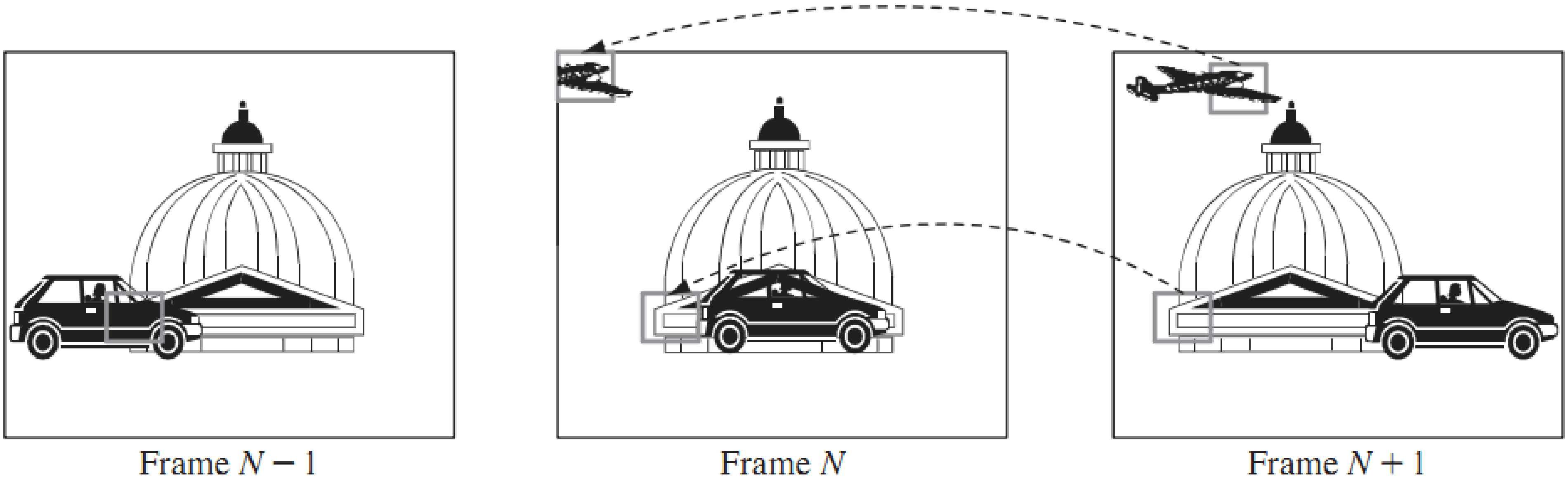
Video Coding Order



Encoding order: 1 4 2 3 7 5 6

Intra: coded directly; **P**redictive: predicted from a previous frame;
Bidirectional: predicted from a previous frame and a following frame.

Benefits of B coding mode



- If there are objects moving into the picture (the air plane above), these new objects cannot be predicted from the previous picture, but can be predicted from the future picture.

Temporal Prediction

- No Motion Compensation:
  - Work well in stationary regions

$$\hat{f}(t,m,n) = f(t-1,m,n)$$

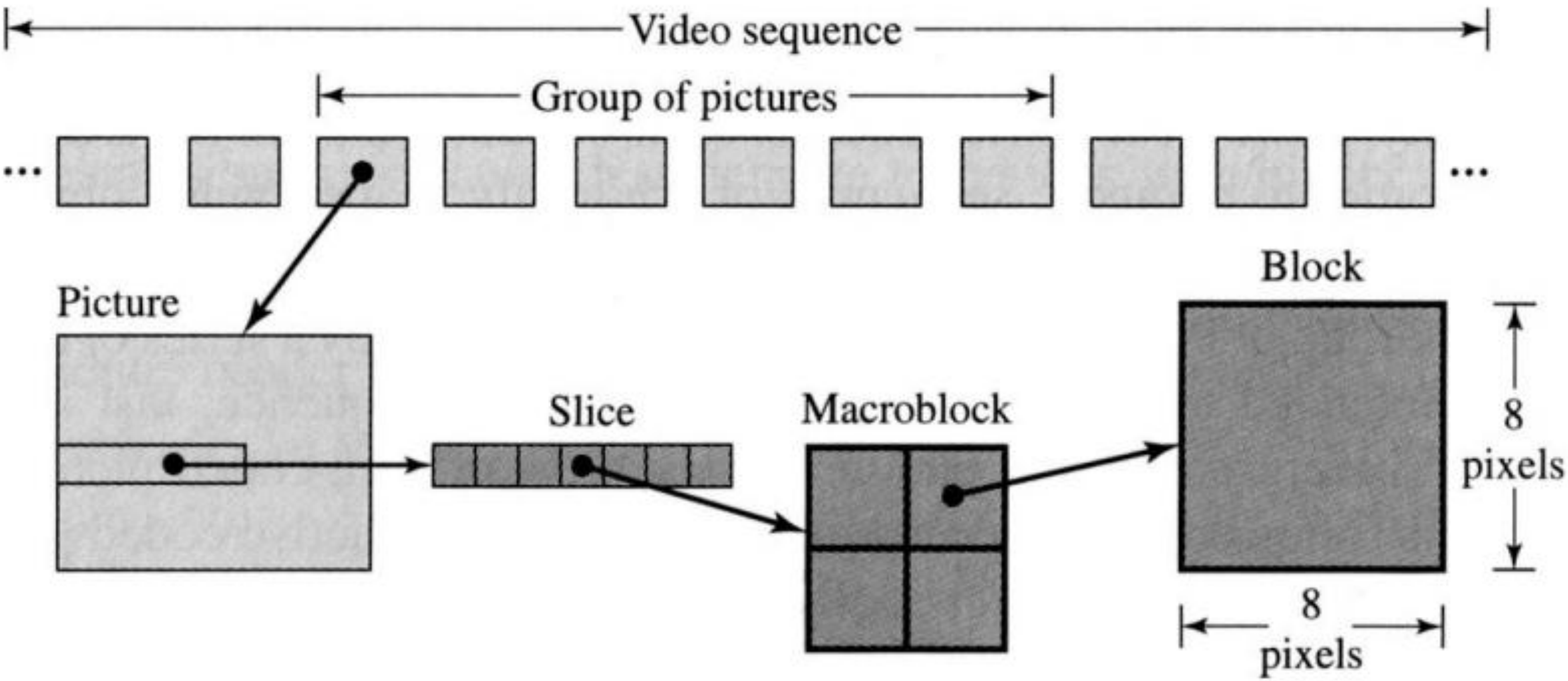
- Uni-directional Motion Compensation:
  - Does not work well for uncovered regions due to object motion or newly appeared objects

$$\hat{f}(t,m,n) = f(t-1,m-d_x,n-d_y)$$

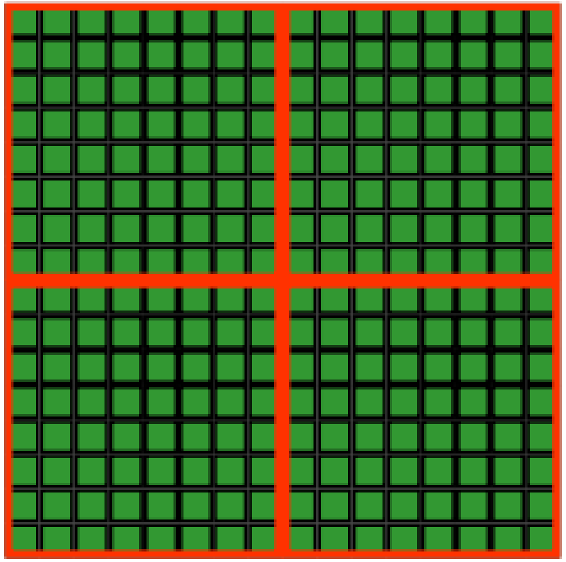
- Bi-directional Motion Compensation
  - Can handle better covered/uncovered regions

$$\hat{f}(t,m,n) = w_b f(t-1,m-d_{b,x},n-d_{b,y}) + w_f f(t+1,m-d_{f,x},n-d_{f,y})$$

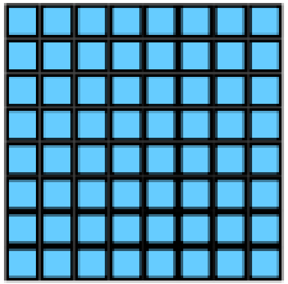
Video Coding Syntax



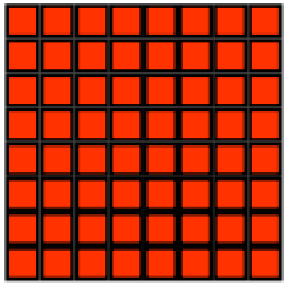
MB Structure in 4:2:0 Format



4 8x8 Y blocks



1 8x8 Cb blocks



1 8x8 Cr blocks

MB Coding in I mode

DCT transform each 8x8 DCT block

↓

Quantize the DCT coefficients with properly chosen quantization matrices (different matrices for Y and C)

↓

The quantized DCT coefficients are zig-zag ordered and run-length coded

MB Coding in P mode

For each macroblock (16x16), find the best matching block in a previous frame, and calculate the prediction errors

↓

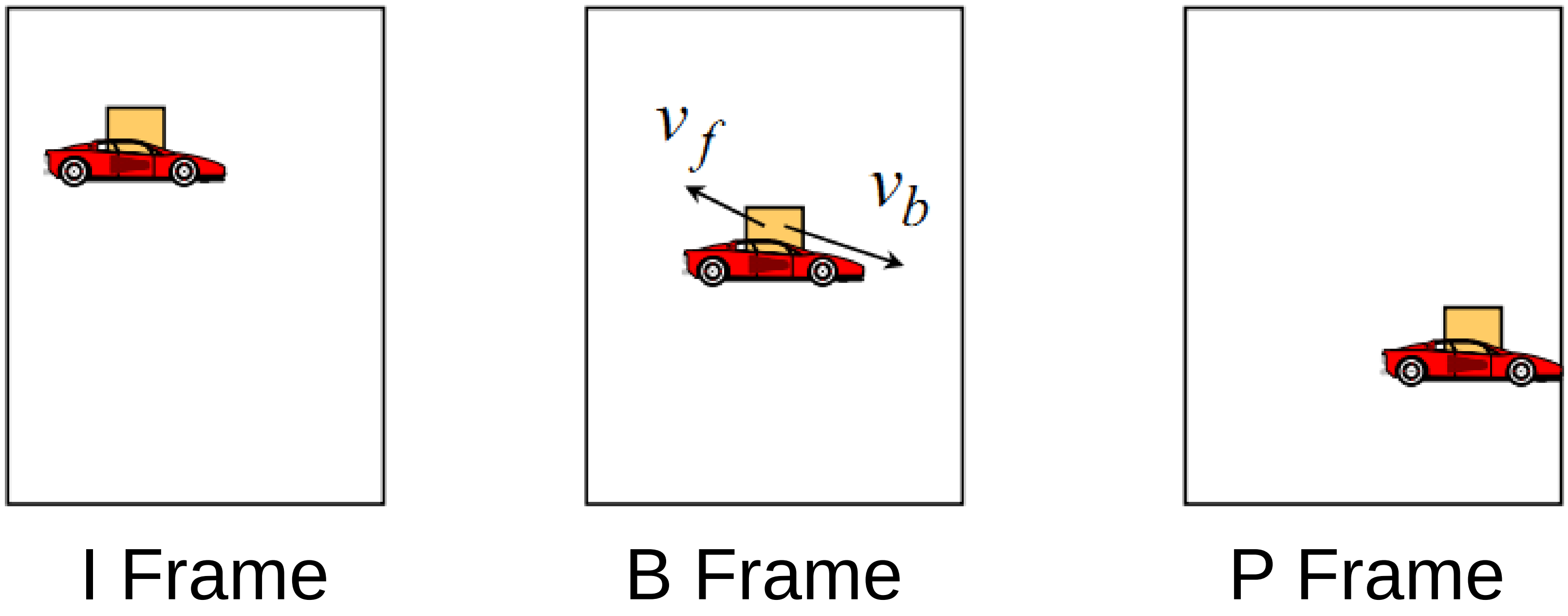
The prediction errors in each of the DCT blocks (8x8) are DCT transformed, quantized (according to specified QP), zig-zag scanned, and run-length coded

↓

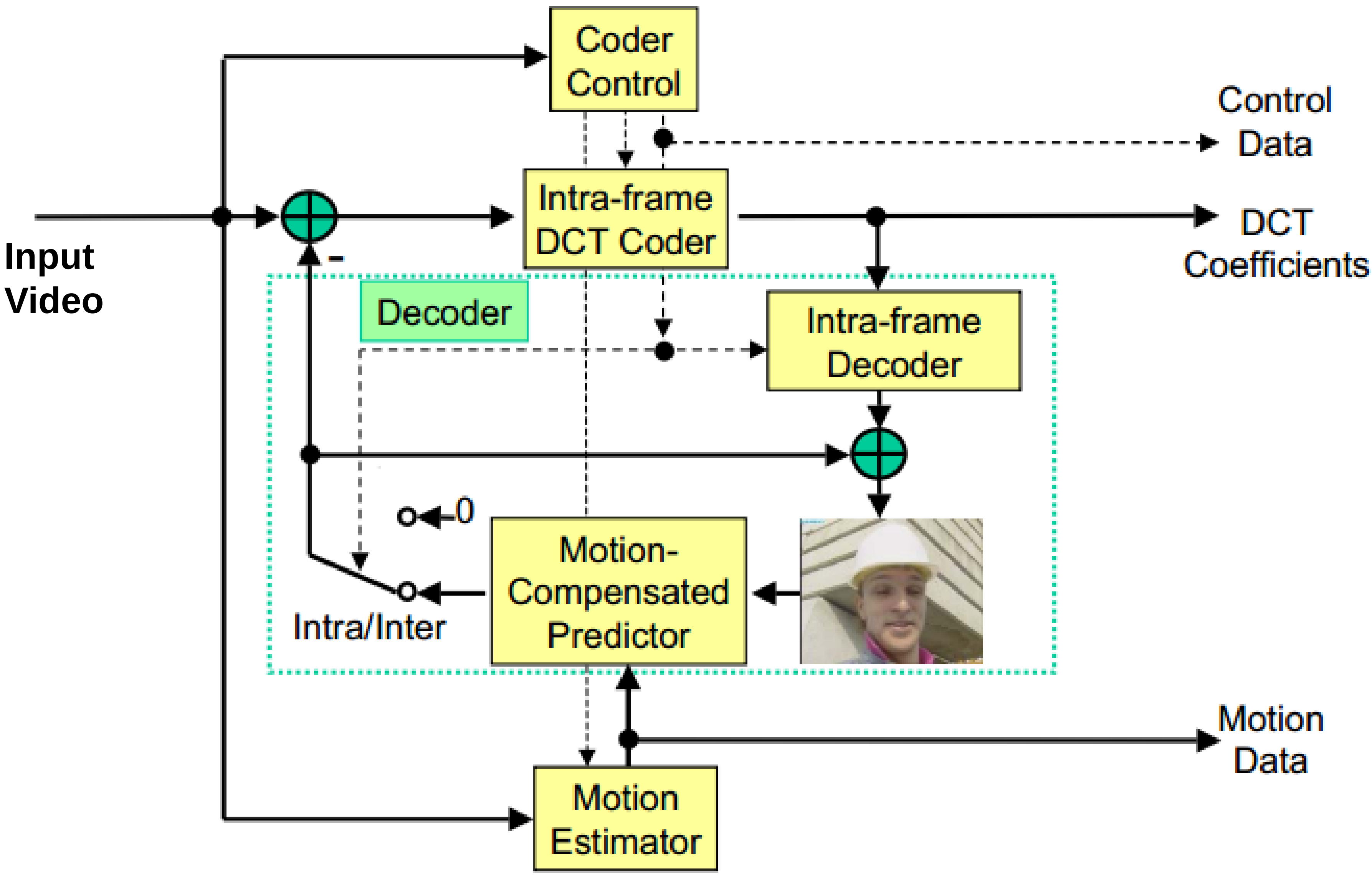
1 pair of motion vector (MV) also needs to be coded

MB Coding in B mode

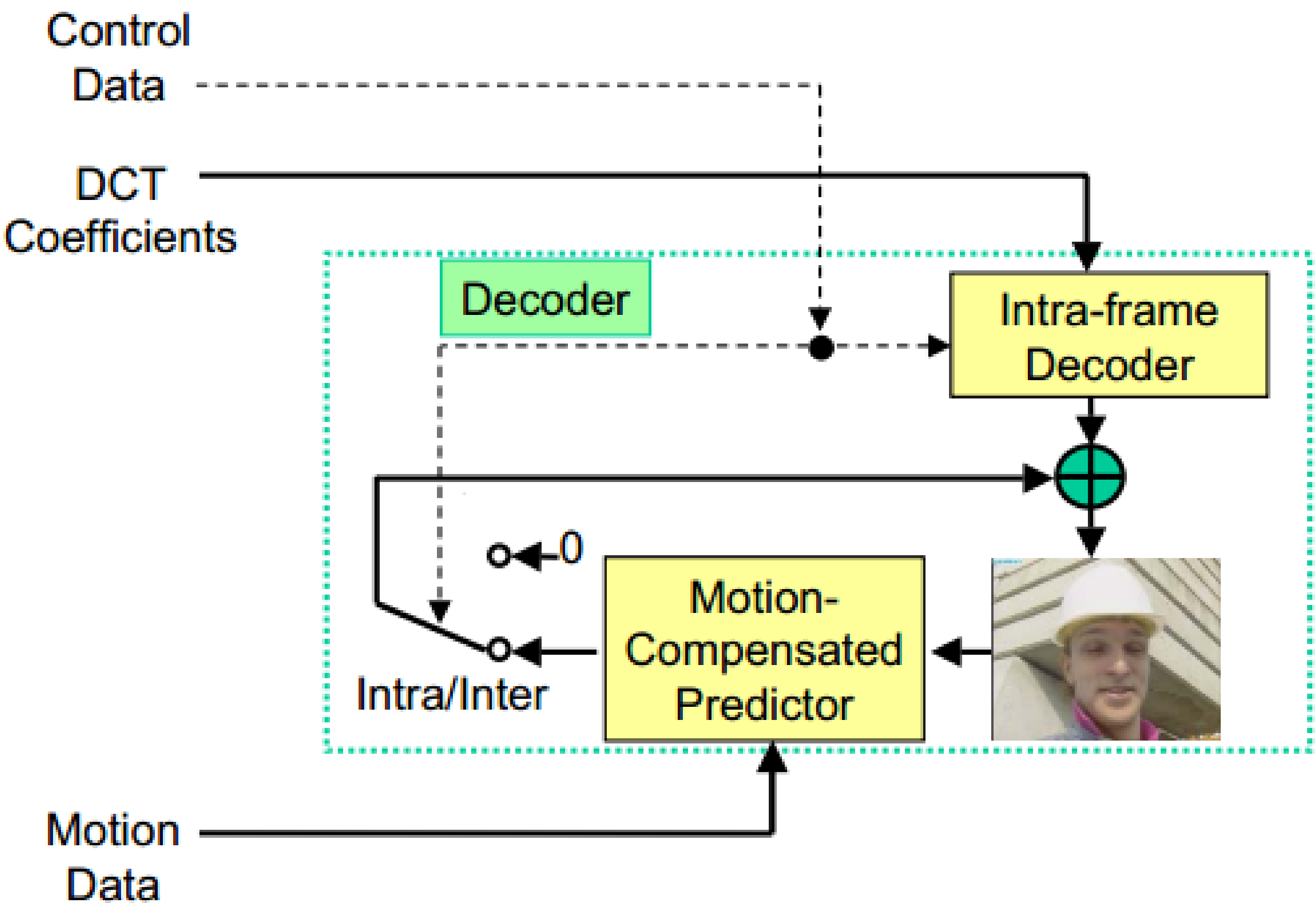
- Same as for the P-mode, except that a macroblock is predicted from both a previous picture and a following one.
- Two pair of MVs needed to be coded.



Block Diagram of Video Encoder



Block Diagram of Video Decoder



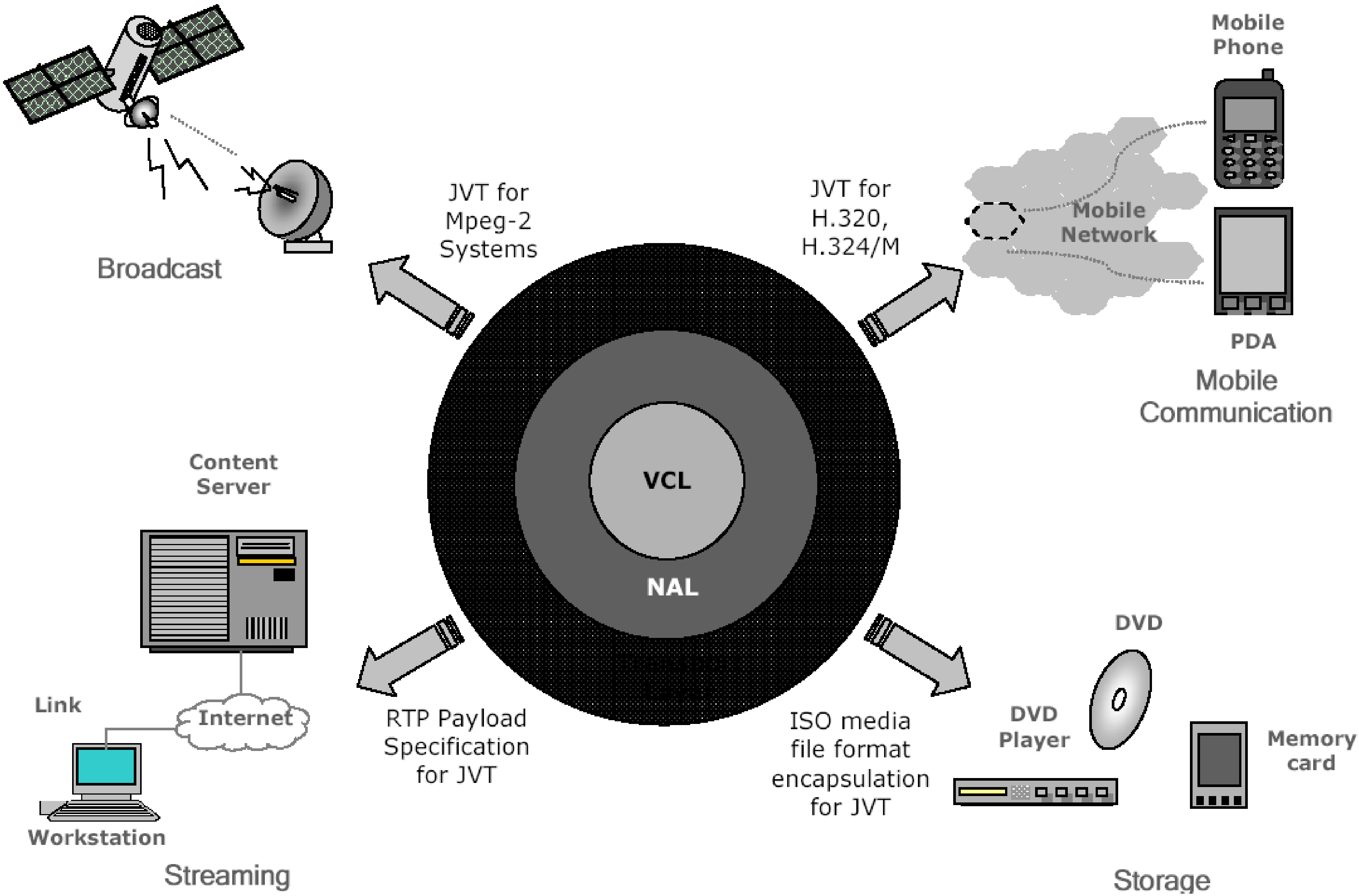
H.264 Video Compression

- Jointly developed by *ITU-T Video Coding Experts Group (VCEG)* and the *ISO/IEC Moving Picture Experts Group (MPEG)* in 2003

- Primary goals:
  - Improved Coding Efficiency
  - Improved Network Adaptation

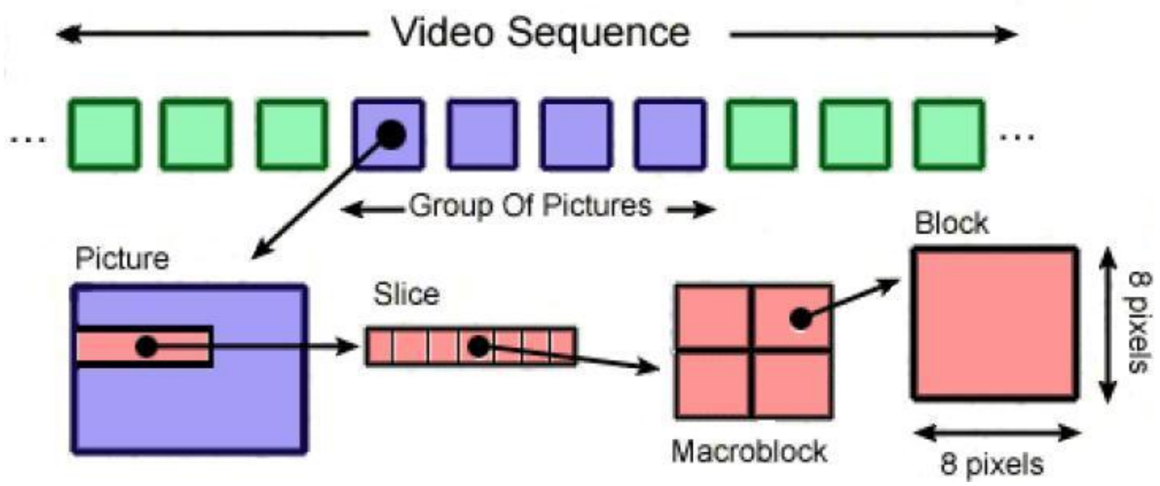
- Application Areas:
  - Conversational services over ISDN & mobile n/ws
  - Storage on CD/DVD
  - IP Video Phones
  - Mobile Multimedia
  - IPTV (Video Streaming over IP networks)

H.264 Eco System

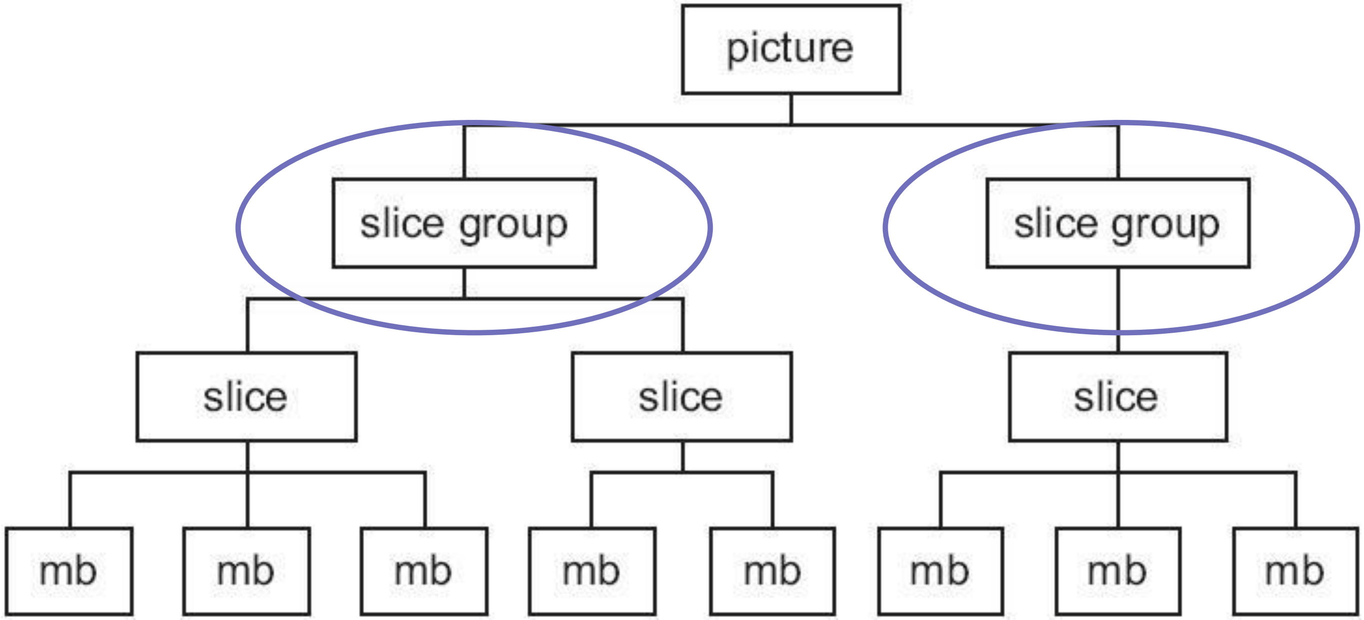


H.264 Slice Groups

Conventional:

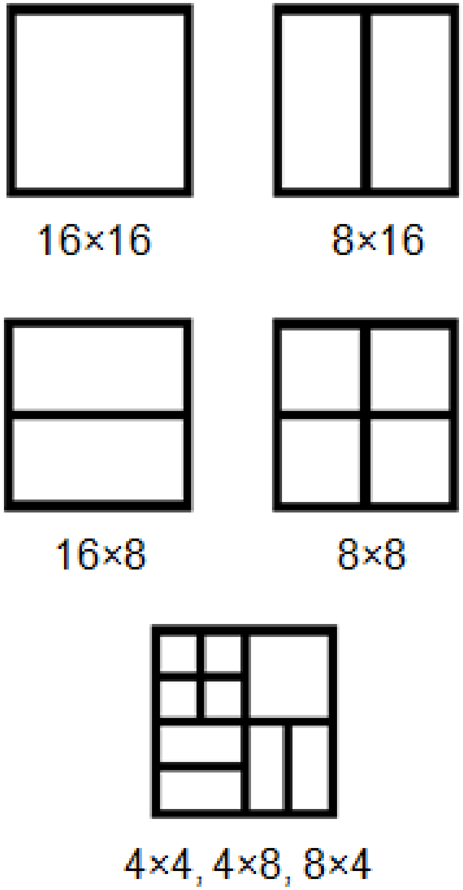


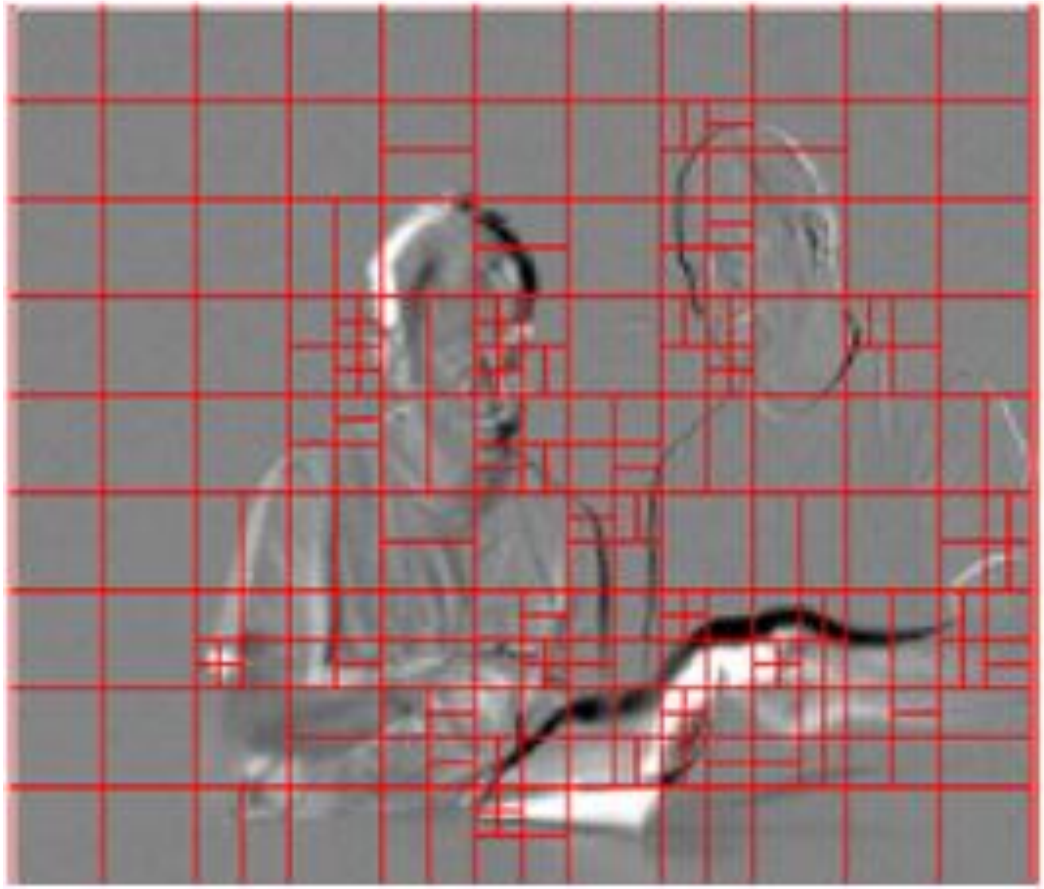
H.264:



H.264 Motion Compensation

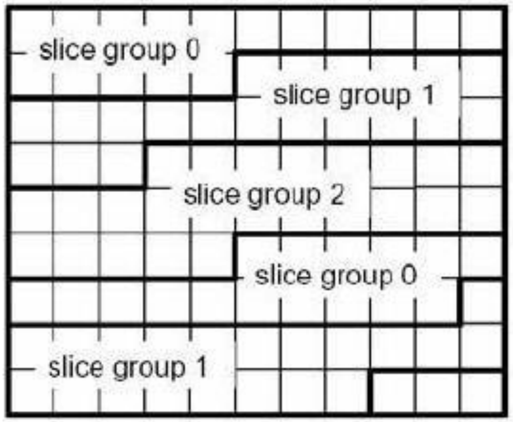
- H.264/AVC uses tree-structured motion compensation with variable block sizes



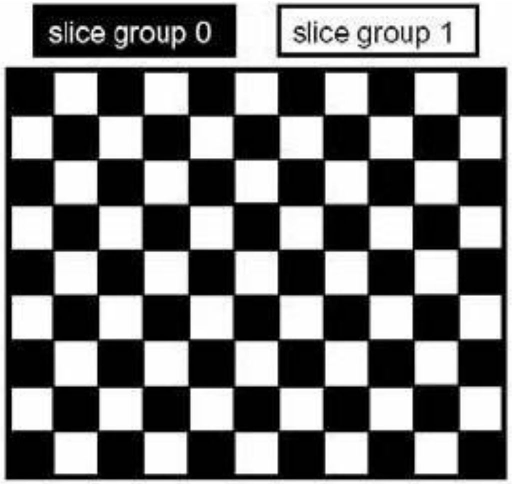


H.264 Error Resilience Tool

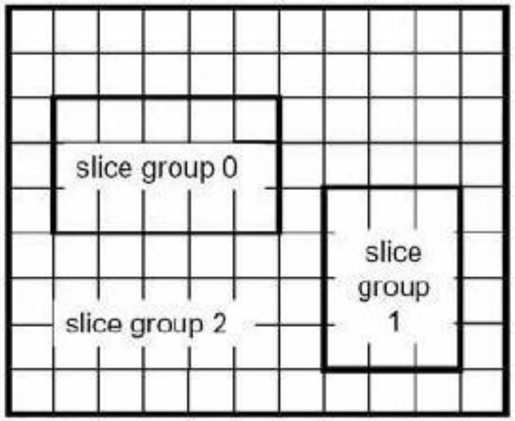
Flexible Macroblock Ordering



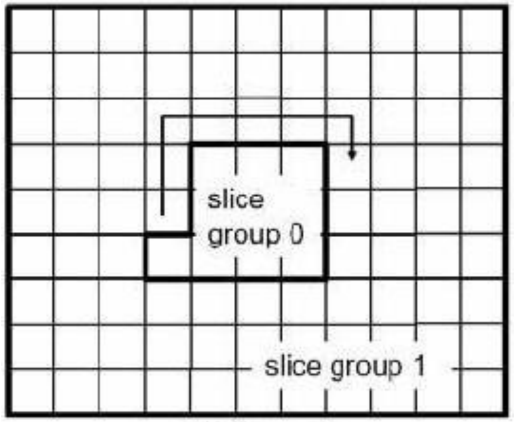
(a) Type 0



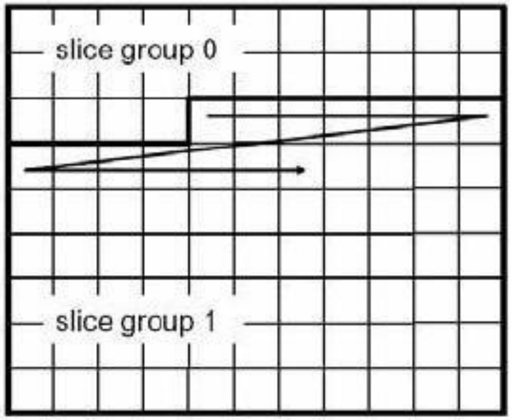
(b) Type 1



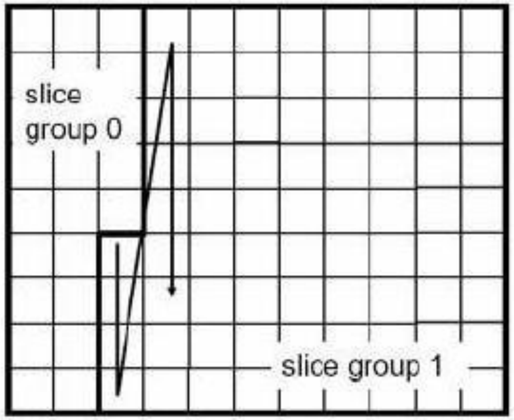
(c) Type 2



(d) Type 3



(e) Type 4



(f) Type 5

Video Streaming: Transmission Errors

Coded, No loss

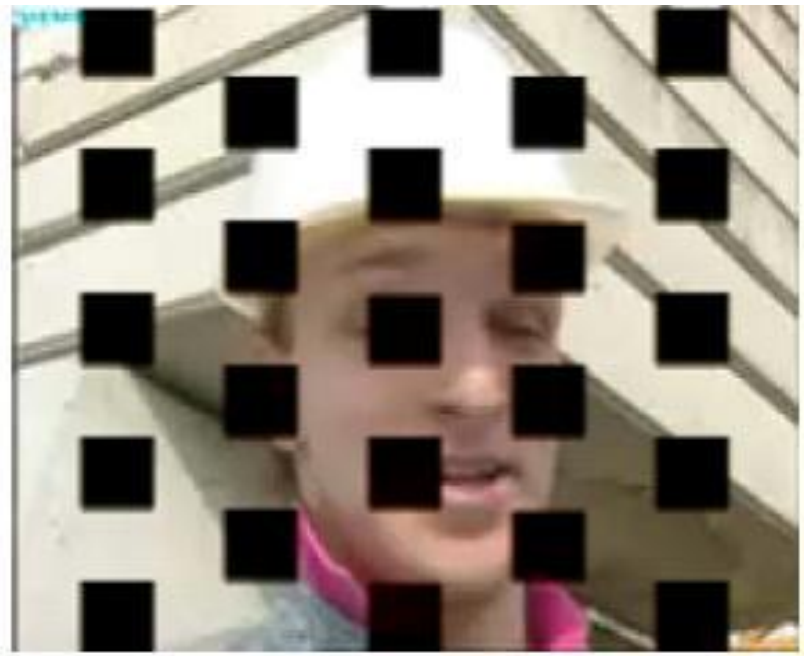
3%

5%

10%

FMO: Effective to Conceal Errors

Slice in Error,
without FMO

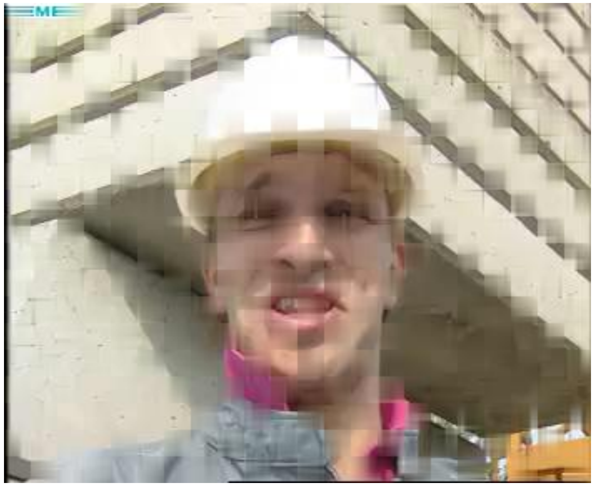


Erroneous Slice can be
recovered due to FMO
Type I (Interleaved)

FMO: Effective to Conceal Errors

Raster scan
@ 10% loss

Interleaved FMO
@ 10% loss



Dispersed FMO
@ 10% loss

Published Results - 1

Santosh Chapaneri, Jeffery Rodriguez, “Low Complexity Error Concealment Scheme for Intra-frames in H.264/AVC”, in Proc. **IEEE International Conference on Image Processing** (ICIP 2009), Egypt, pp. 925-928, Nov 2009

Foreman CIF Intra frame @ 10% loss rate

Concealed with Xu’s method
PSNR = 33.17 dB, MSSIM = 0.8035

Concealed with PMEC
PSNR = 34.26 dB, MSSIM = 0.9126

Published Results - 2

Santosh Chapaneri, Jeffery Rodriguez, “Content-Adaptive Macroblock Partitioning Scheme for Error Concealment of H.264/AVC Coded Video”, in Proc. **IEEE International Conference on Image Processing** (ICIP 2009), Egypt, pp. 917-920, Nov 2009

Stefan CIF Inter frame @ 10% loss rate

Concealed with ABS
PSNR = 26.35 dB, Q = 0.8726

Concealed with CAMP
PSNR = 27.63 dB, Q = 0.9432

Published Results - 3

Santosh Chapaneri, “Content-Adaptive Refined Error Concealment Schemes for H.264/AVC Video Coding”, in **International Journal of Computer Applications**, Vol. 27, No. 7, pp. 36-43, Aug 2011

Table-tennis QCIF Inter frame @ 10% loss rate

Concealed with Xu’s method (RSTC)
PSNR = 28.02 dB, Q = 0.8179

Concealed with CAREC
PSNR = 29.85 dB, Q = 0.8842

Region of Interest (ROI) Coding

- FMO Type 2 can be used to distinguish between foreground and background regions
- Foreground region can be defined as the Region of Interest changing **dynamically** and coded with high quality, thus **preserving the important details** of the video sequence
- **ROI can be obtained through a video segmentation algorithm** for each picture using spatial, color and/or motion cues
- A new Picture Parameter Set (PPS) is created every time the ROI changes

ROI using FMO

(a) Frame 0

(b) Frame 6

(c) Frame 12

(d) Frame 34

(e) Frame 65

(f) Frame 95

ROI using FMO @ 5% packet loss

Frame 240 without ROI/FMO

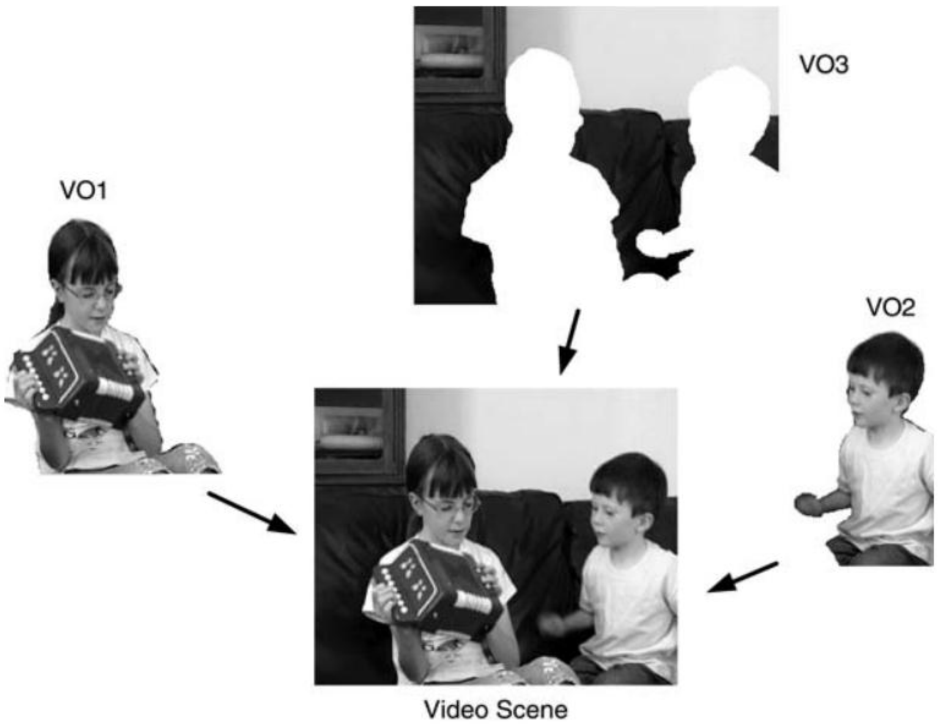
Frame 240 with ROI/FMO

Frame 270 without ROI/FMO

Frame 270 with ROI/FMO

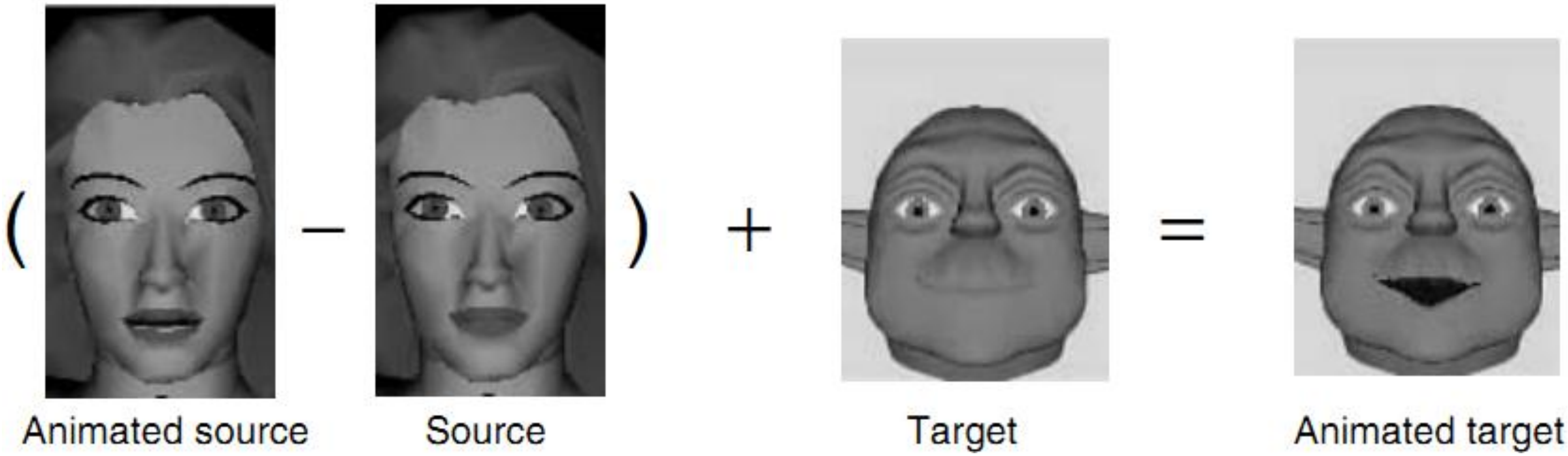
MPEG-4 Video Standard

- MPEG-4 treats a video sequence as a collection of one or more **video objects**.
- A video object (VO) is an area of the video scene that may occupy an arbitrarily-shaped region and may exist for an arbitrary length of time.
- An instance of a VO at a particular point in time is a video object plane (VOP)



MPEG-4 Video Standard

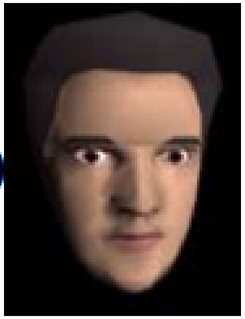
- MPEG-4 also supports **Face and Body Animation** using Synthetic Video Coding



MPEG-4 can create Avatars

Synthetic Objects in MPEG-4

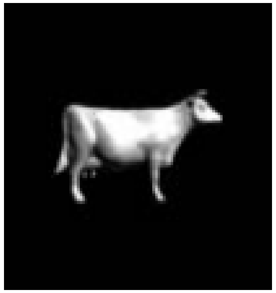
- Face coding and its animation



- Body coding and its animation

- Photo-realistic 3D model coding

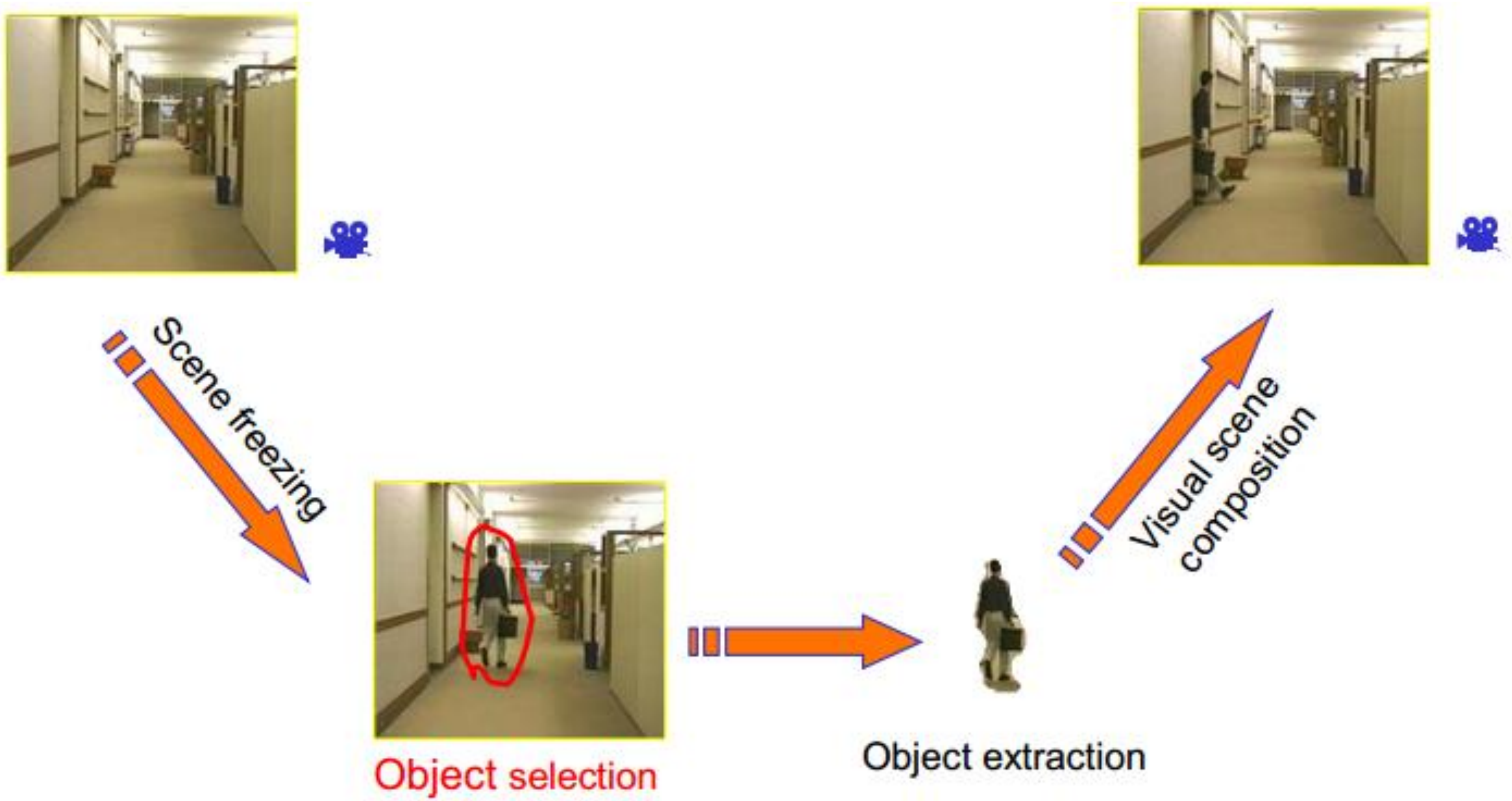
- Animated 3D model coding



Application: Augmented Reality

Application: Tele-Presence

Application: Video Editing



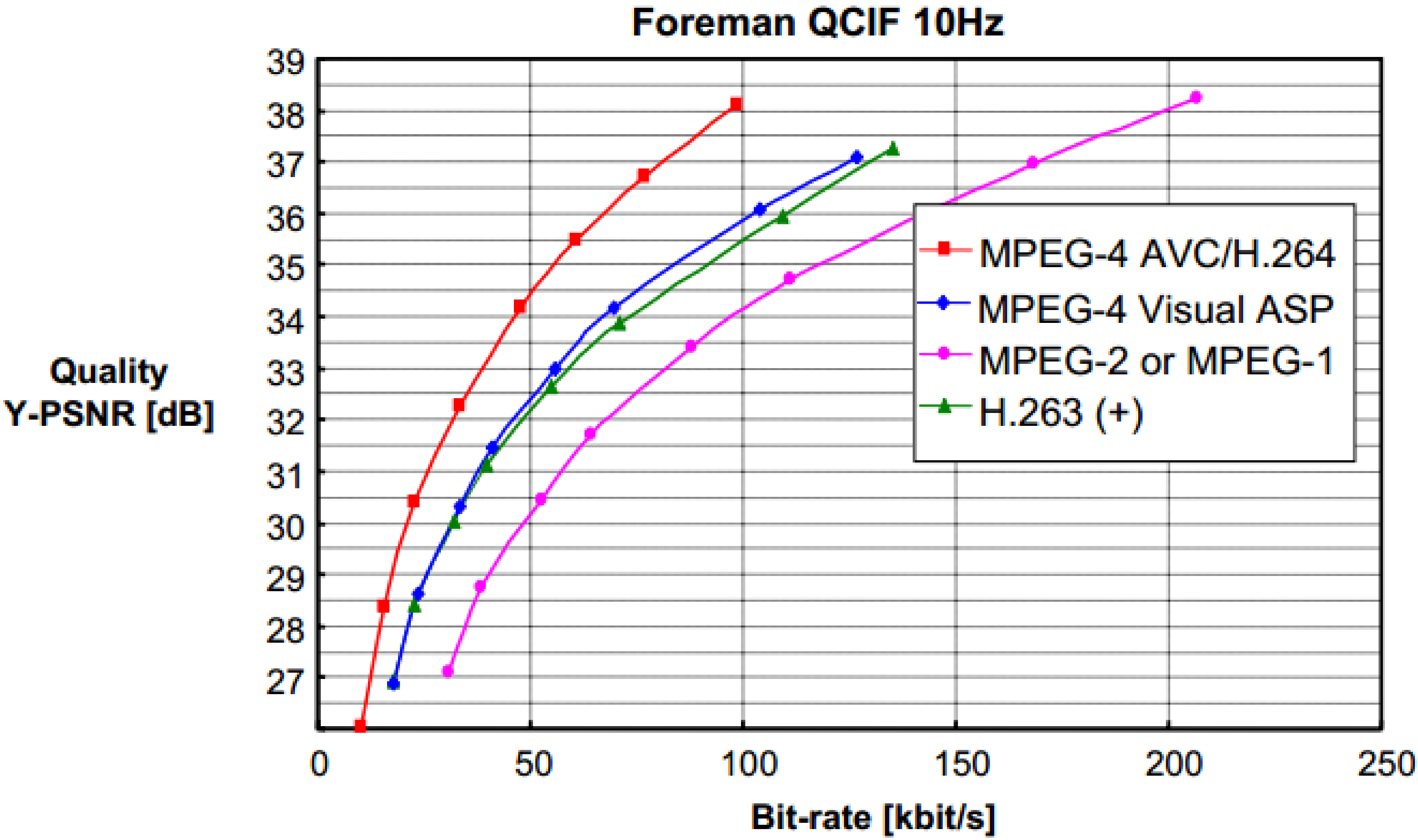
Some MPEG-4 Products

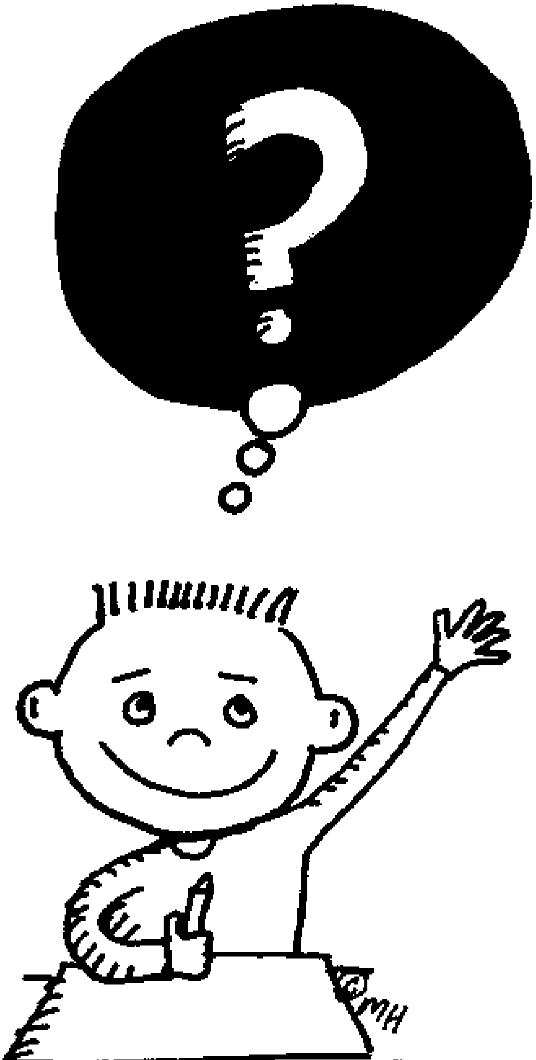
- Media Players (Microsoft, QuickTime, …)
- Video (surveillance) cameras (Sharp, Sanyo, Cisco, …)
- Mobile audiovisual codecs (UMTS)

Internet ViewCam
<VN-EZ1>

Scalable Video Coding

Performance Comparison





QUESTIONS