10. MPEG-1: Video Compression

10.1 Introduction

- **MPEG-1…**
  - The first audiovisual coding standard from the ISO **M**oving **P**icture **E**xperts **G**roup
  - Improves on H.261 (earlier standard for video conferencing)
  - Target → Delivery of compressed video at the *maximum data-transfer rate* handled by CD-ROM drives at the time
    - ~ 1.5Mbps

10.1 Introduction

- **MPEG-1…**
  - The first audiovisual coding standard from the ISO **M**oving **P**icture **E**xperts **G**roup
  - Improves on H.261 (earlier standard for video conferencing)
  - Target → Delivery of compressed video at the *maximum data-transfer rate* handled by CD-ROM drives at the time
    - ~ 1.5Mbps
- **As a counterpart to JPEG…**
  - Exploits both spatial and perceptual redundancy in digital images
    - But also the temporal redundancy between successive frames
- **Note** → A *decoding* standard only
    - i.e. bitstream format the only aspect standardised

10.1.1 MPEG-1 Standards Documents

- **MPEG-1 is a standard in five parts…**
  - Part-1 (Systems: ISO/IEC 11172-1: 1993)
    - Addresses the problem of combining one or more data streams from the video and audio parts of the MPEG-1 standard with timing information to form a single stream, i.e. multiplexing and synchronization of audio/video data. Once combined into a single stream, the data is well suited to digital storage and transmission.
  - **Part-2 (Video: ISO/IEC 11172-2: 1993)**
    - Specifies a coded representation that can be used for compressing video sequences to the principal target bit-rate of 1.5Mbps. However, since the approaches undertaken are generic in nature, the standard may be used more widely than the specified bitrate.
  - Part-3 (Audio: ISO/IEC 11172-3: 1993)
    - Specifies a coded representation that can be used for compressing audio sequences. A psycho-acoustic model creates a set of data to control the quantifier and coding.
  - Part-4 (Conformance Testing: ISO/IEC 11172-4: 1995)
    - Specifies how tests can be designed to verify whether bit-streams and decoders meet the requirements as specified in parts 1, 2 and 3 of the standard.
  - Part-5 (Software Simulation: ISO/IEC TR 11172-5)
    - Technically not a standard, but rather a technical report. It provides a full software implementation of the first three parts of the MPEG-1 standard (source code is not publicly available).

10. MPEG-1: Video Compression

10.2 Overview of MPEG-1 Video

- **MPEG-1: Main Features…**
  - Targets CD-ROM-based multimedia applications (~ 1.5Mbps)
  - Supports flexible picture size and flexible frame-rate
  - Retains much of the ground work employed in H.261
  - Supports picture-based random access of video
  - Supports FF/FR through the compressed stream
  - Supports reverse playback of video
  - Supports editability of the compressed stream

10.2 Overview of MPEG-1 Video (cont.)

- **MPEG-1 key elements…**
  - **Motion Compensation**
    - A block (‘*macroblock*’) matching technique to account for moving objects in the scene (see other Chapter)
  - **Discrete Cosine Transform (DCT)**
    - Transforms the 2-D image data towards increased ‘energy compaction’
  - **Quantization**
    - High freq DCT coeffs coded more coarsely
  - **Entropy Coding**
    - Fixed Huffman codes used to code quantized DCT coeff ‘events’
      - e.g. (*length*, *magnitude*) events for DCCs; (*run/length*, *magnitude*) events for ACCs

As in JPEG!

10.2 Overview of MPEG-1 Video (cont.)

The idea of motion estimation is not to describe object(s) motion trajectory!

It is used to reduce number of details (motion compensation).

Frame 69

Predicted Frame69

Absolute Difference w/o Motion Compensation

Absolute Difference with Motion Compensation

10.2 Overview of MPEG-1 Video (cont.)

- **MPEG-1 Picture Types (3 main types)**
  - **I-picture**
    - Coded independently (similar to JPEG)
    - Provide access points (random stream access)
    - Facilitate FF/FR functionality (I-frames displayed when skimming)
    - Relatively bit-rate intensive (NO temporal redundancy exploited!)
  - **P-picture**
    - Coded as prediction from a previous picture (either I or P)
  - **B-picture**
    - Coded as combined prediction from both a previous picture (I or P) AND a future picture (I or P) (→ yields high compression!)
    - Never used as a reference for prediction!

      → Hence can be coded 'lossily' without error propagation ☺

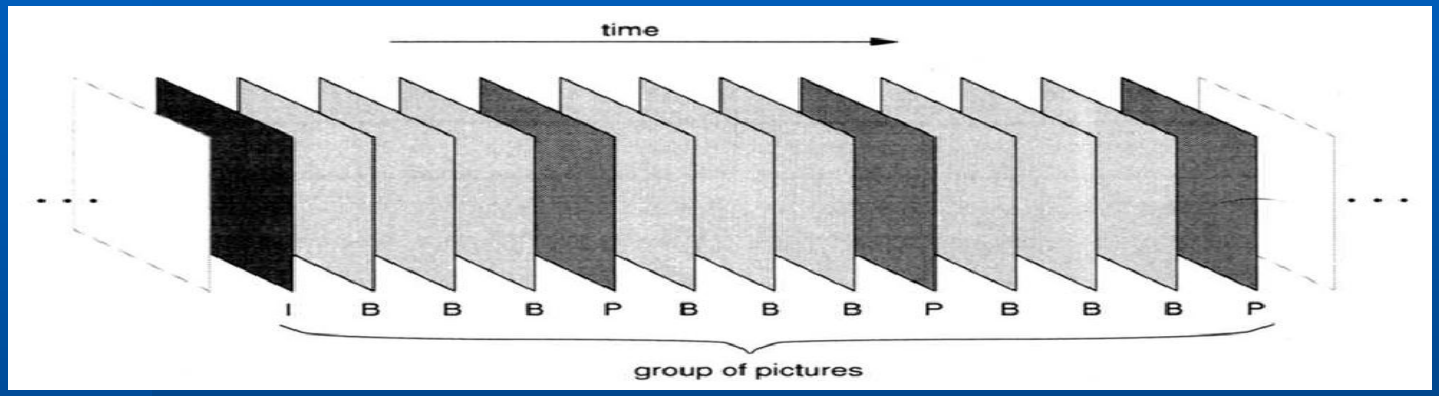
I

P

B

10.2 Overview of MPEG-1 Video (cont.)

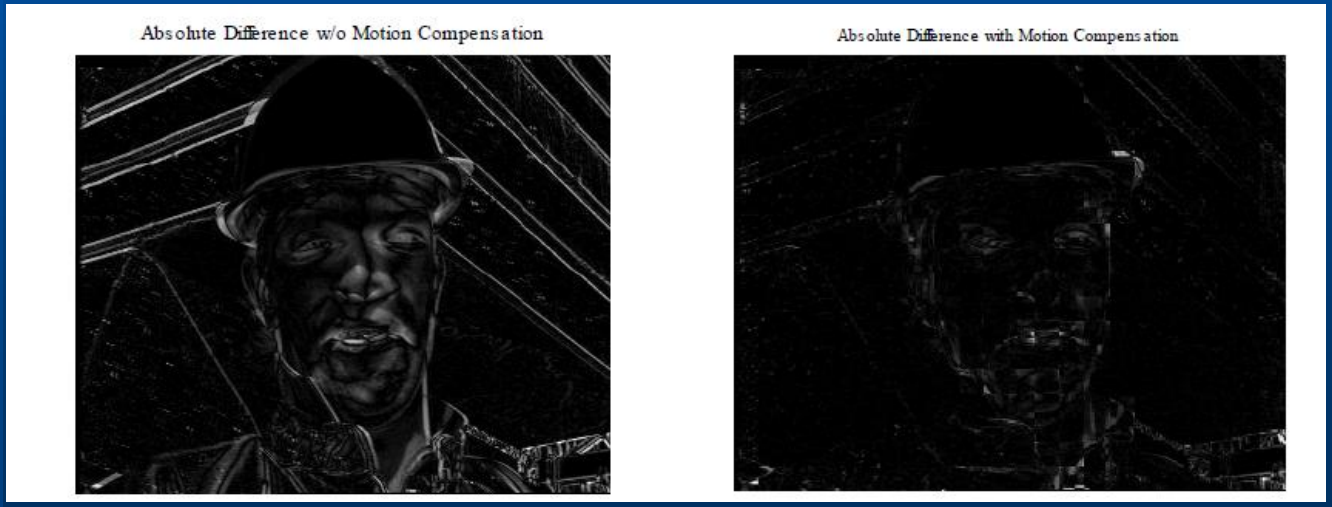
- **MPEG-1 Picture Types: Temporal relationship…**



  - I-pic coded independently → first P-pic predicted from I-pic
  - 3 intervening B-pics predicted from combination of I-pic & first P-pic
    - Combinations equally weighted irrespective of proximity to reference pic
  - Second P-pic predicted from first P-pic
  - Second group of three intervening B-pics predicted from combination of first and second P-pics
  - Etc.

10.2 Overview of MPEG-1 Video (cont.)

- **What is meant by 'prediction'?**
  - '***Target image***' subtracted from '***Reference image***'
  - '***Residual image***' then DCT encoded
    - N.B. If target ≈ reference…
      - Residual will be small, and should be quantized to zero! ☺
  - Note → prediction actually performed at the '***macroblock***' level



10.2 Overview of MPEG-1 Video (cont.)

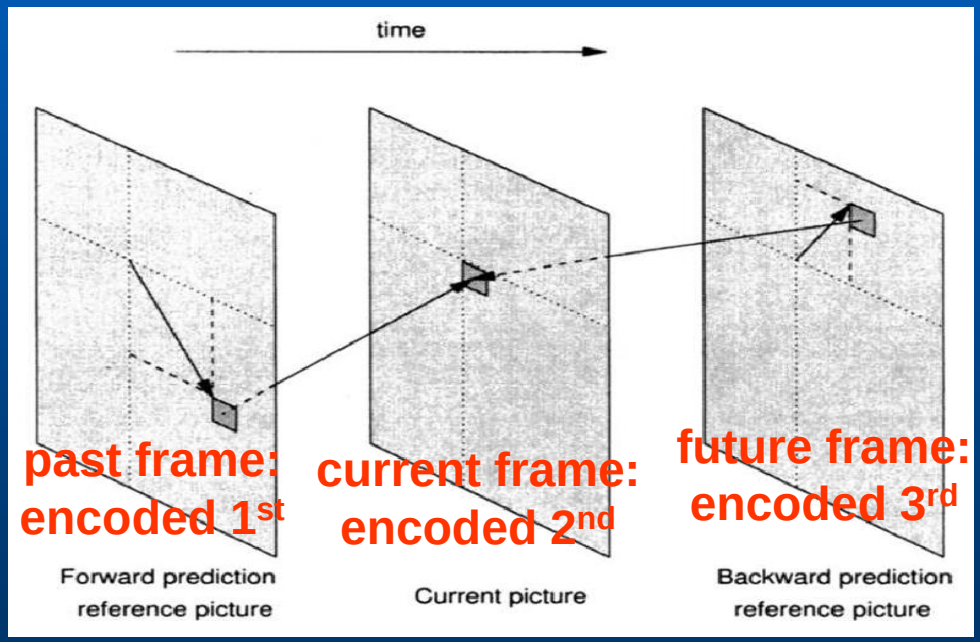
- **What is meant by 'prediction'?**
  - '***Target image***' subtracted from '***Reference image***'
  - '***Residual image***' then DCT encoded
    - N.B. If target ≈ reference…
      - Residual will be small, and should be quantized to zero! ☺
  - Note → prediction actually performed at the '***macroblock***' level

- **Macroblocks? (same as H.261)**
  - DCT performed at block level (8x8 pixels)
  - Macroblock (mblk) ≡ 16x16 block of pixels (four 8x8 'quadrants')
    - Image prediction (subtraction) performed at this level
      - Trade-off between increasing coding efficiency (smaller regions yielding smaller residuals) and the decreased overhead in having fewer regions (motion vectors)

10.2 Overview of MPEG-1 Video (cont.)

ILLUSTRATION: Bi-directional prediction in B-pics

- Highlighted mblk to be predicted based on reference mblks from past & future frames (I or P)
- Reference mblks are specified by '***Motion Vectors***'
  → indicate relative displacements from the *target* mblk
  → allows positions of reference mblks to vary
  → hopefully end up choosing those offering smallest residual! ☺
- N.B. zero motion vectors OK too!



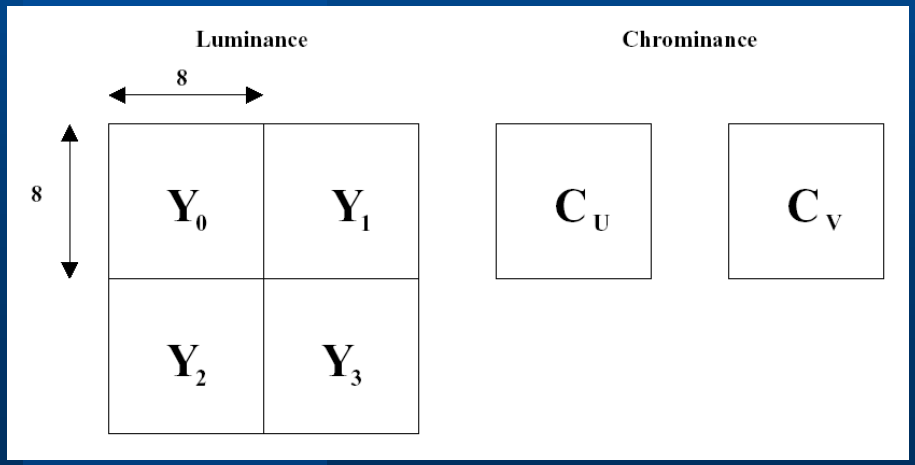
10.2 Overview of MPEG-1 Video (cont.)

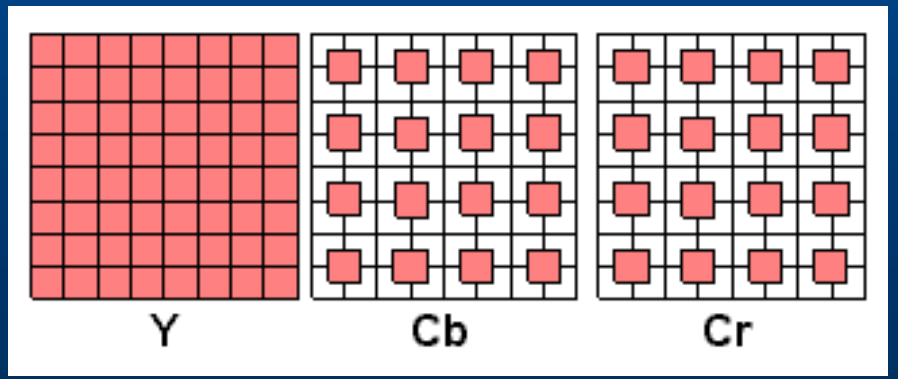
MPEG-1 Colour Space

- JPEG: YUV colour space used (reduced inter-component correlation)

  → Allows exploitation of HVS's relative insensitivity to chrominance
  - U/V signals subsampled relative to Y
- Similarily in MPEG: UV signals subsampled

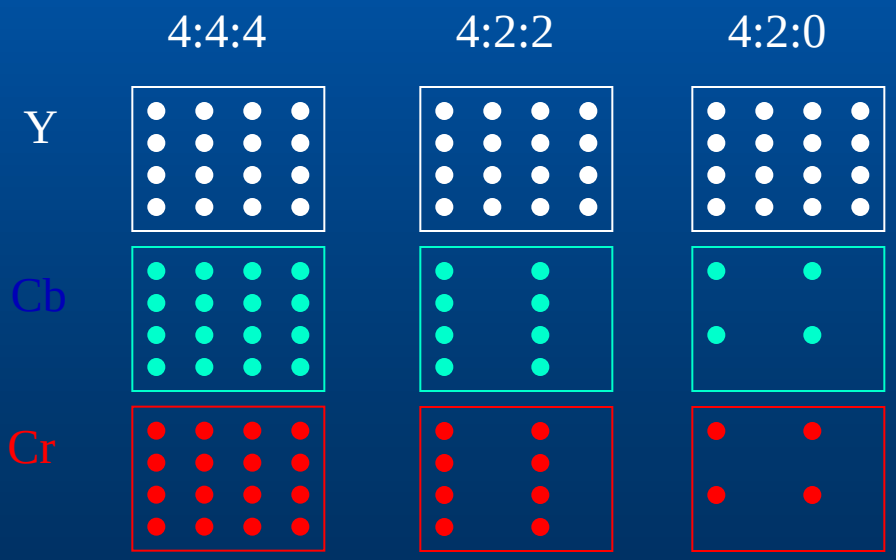
  → Each 16x16 mblk in a video image consists of four 8x8 Y blocks, one 8x8 U block, one 8x8 V block → overlaid as shown…





10.2 Overview of MPEG-1 Video (cont.)

- **Human Visual System more sensitive to luma**
  - **Chroma frequently sub-sampled**
  - **Sub-sampling examples**



10. MPEG-1: Video Compression

10.3.1 Intra Coding: I-frames

- **I-pictures, how often?**
  - 1st frame in sequence always intra-coded
  - Application requirements → determines the 'gap' between I-pics
    - MPEG-1 standard: max gap = 128 frames
- **Steps for Intra Coding…**
  - For first (16x16) mblk of image …
    - 8x8 DCT applied to its six 8x8 blocks: Y0, Y1, Y2, Y3, U, and V
    - DCT coeffs are then uniformly quantized (equal width bands) by…
      - Dividing each coeff by assigned 'step-size' from QT
        - Remainder discarded
      - QT retained for reconstruction

Same as JPEG!

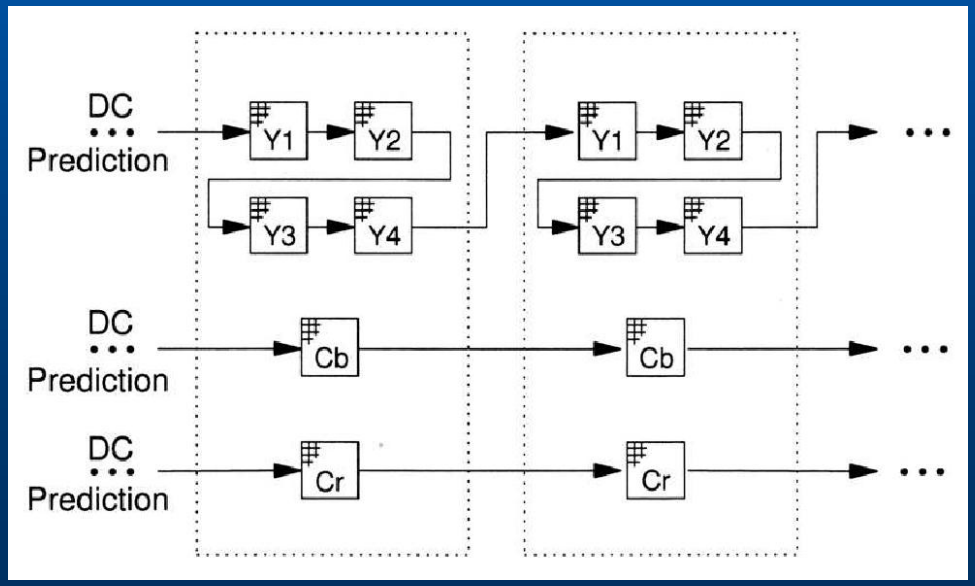
8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

10.3.1 Intra Coding: I-frames (cont.)

- **Steps for Intra Coding (cont.)…**
  - Y-DCCs predicted (differenced) across ALL Y-blocks of the IMAGE
  - U & V DCCs predicted across ALL U & V blocks of the IMAGE
  - ACCs of each block (4xY, 1xU, 1xV)…
    - Zig-zag scanned
    - Run-length encoded
    - Entropy coded

Same as JPEG!

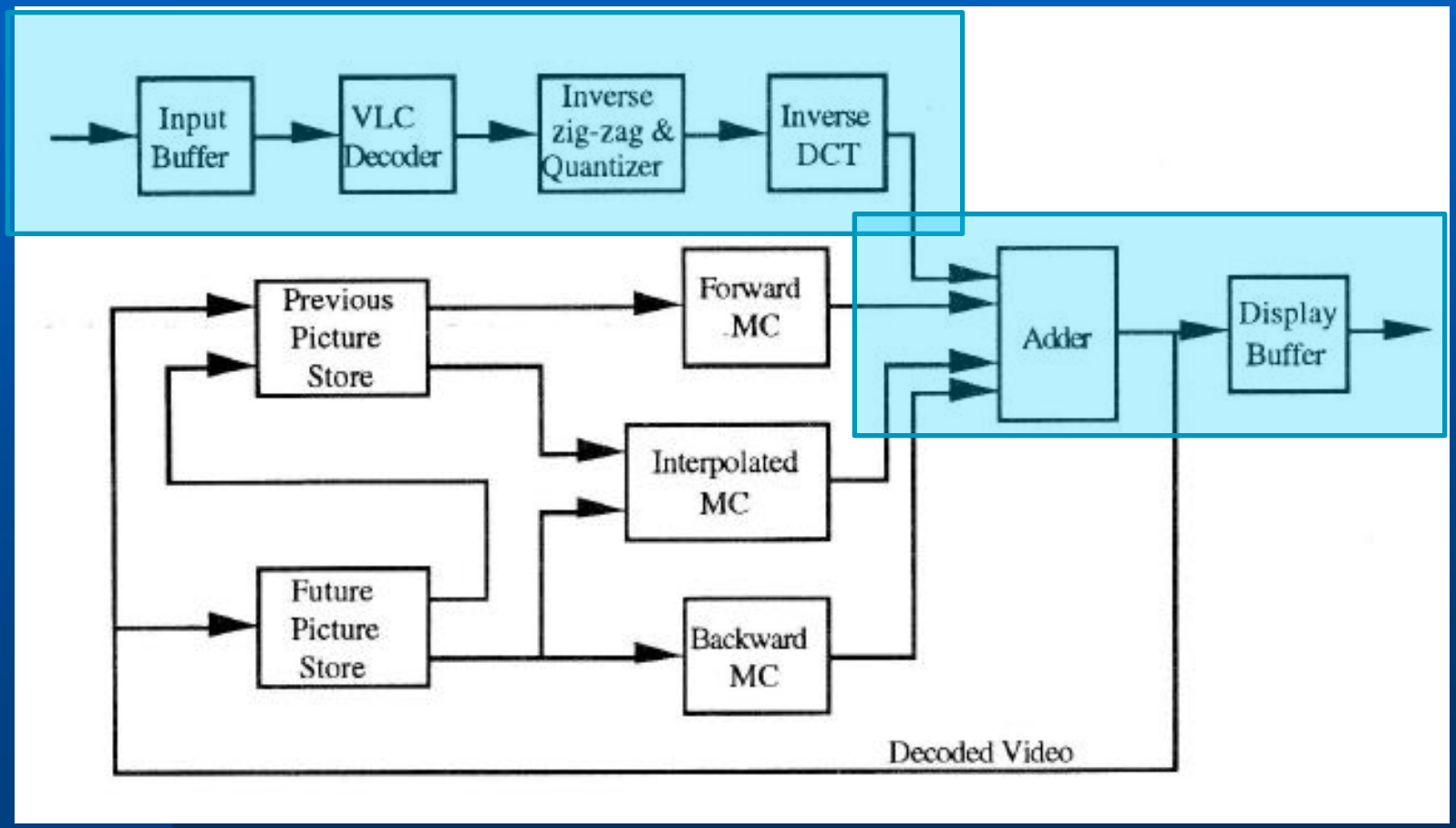
  - Then move on to next mblk!



10.3.1 Intra Coding: I-frames (cont.)

- **Decoding I-pictures**
  - For first mblk of image…
    - Entropy codes (VLCs) & run-length codes deciphered
      → Decoder then knows locations of the non-zero DCT coeffs of each block of the mblk (4xY, 1xU, 1xV)
    - Prediction undone
    - DCT coeffs reconstructed by multiplying by corresponding 'step-size' from QT
    - IDCT performed on the reconstructed coeffs
      → Gives reconstructed Y1, Y2, Y3, Y4, U, V pixel values
  - Then move onto next mblk

Decoding loop structure (I-picture)



Quantization Example

8x8 DCT coeff matrix

$$\begin{bmatrix} -415 & -33 & -58 & 35 & 58 & -51 & -15 & -12 \\ 5 & -34 & 49 & 18 & 27 & 1 & -5 & 3 \\ -46 & 14 & 80 & -35 & -50 & 19 & 7 & -18 \\ -53 & 21 & 34 & -20 & 2 & 34 & 36 & 12 \\ 9 & -2 & 9 & -5 & -32 & -15 & 45 & 37 \\ -8 & 15 & -16 & 7 & -8 & 11 & 4 & 7 \\ 19 & -28 & -2 & -26 & -2 & 7 & -44 & -21 \\ 18 & 25 & -12 & -44 & 35 & 48 & -37 & -3 \end{bmatrix}$$

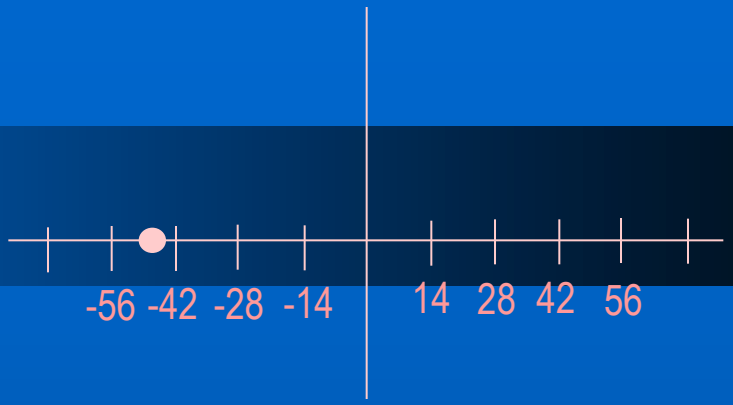
Standard quantization table

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Resulting quantized coeffs

$$\begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -3 & 4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -4 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Quantization Example



8x8 DCT coeff matrix

$$\begin{bmatrix} -415 & -33 & -58 & 35 & 58 & -51 & -15 & -12 \\ 5 & -34 & 49 & 18 & 27 & 1 & -5 & 3 \\ -46 & 14 & 80 & -35 & -50 & 19 & 7 & -18 \\ -53 & 21 & 34 & -20 & 2 & 34 & 36 & 12 \\ 9 & -2 & 9 & -5 & -32 & -15 & 45 & 37 \\ -8 & 15 & -16 & 7 & -8 & 11 & 4 & 7 \\ 19 & -28 & -2 & -26 & -2 & 7 & -44 & -21 \\ 18 & 25 & -12 & -44 & 35 & 48 & -37 & -3 \end{bmatrix}$$

Standard quantization table

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Resulting quantized coeffs

$$\begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -3 & 4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -4 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

E.g. **-46** ÷ 14

= -3.2857

→ rounds to -3

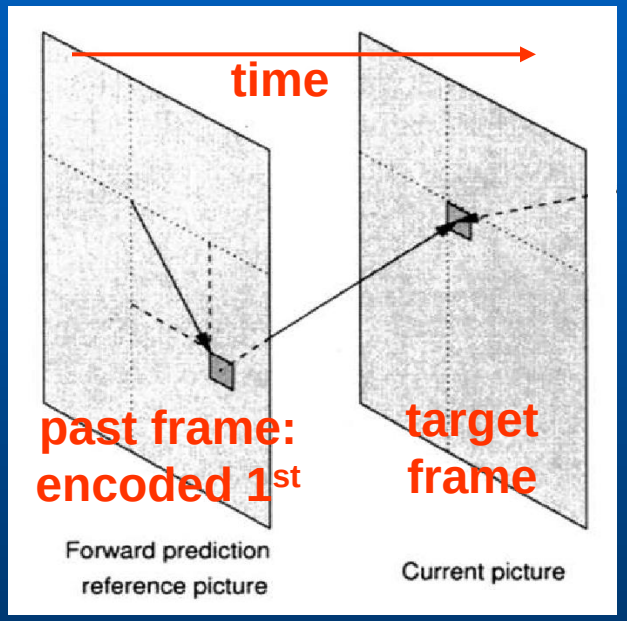
E.g. reconstruction:

-3 x 14 = **-42**

10.3.1 Inter Coding: P/B-frames

Encoding P-pictures (predicted from past I/P-pic)

- For each mblk in the *target* image…
  - Motion Estimation
    → Ref image searched for best 16x16 match (→ motion vectors)
    → N.B. match does not have to align to mblk boundaries
  - Target/reference mblks differenced
  - Residual transformed by the 6 DCTs
  - Quantization, RLE, Entropy encoding of coeffs
  - N.B. DCCs not predicted in P/B frames
    → Already represent a differential value!
  - Each processed mblk transmitted along with motion vectors

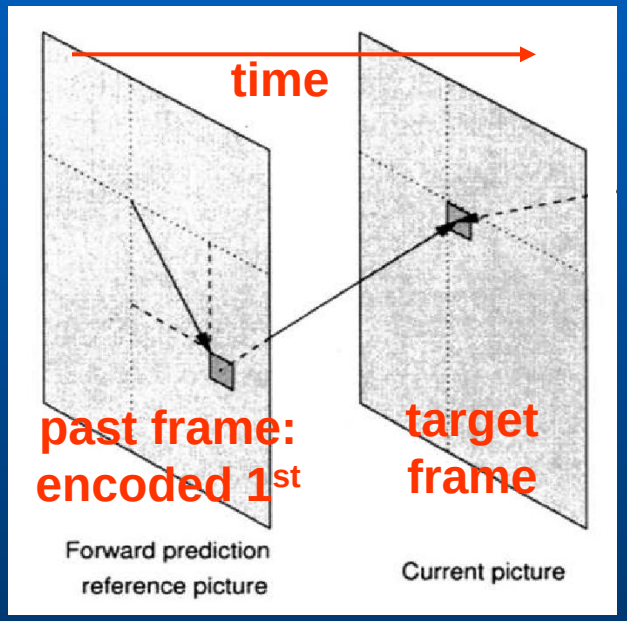


10.3.1 Inter Coding: P/B-frames

Encoding P-pictures (predicted from past I/P-pic)

- For each mblk in the *target* image…
  - Motion Estimation
    → Ref image searched for best 16x16 match (→ motion vectors)
    → N.B. match does not have to align to mblk boundaries
  - Target/reference mblks differenced
  - Residual transformed by the 6 DCTs
  - Quantization, RLE, Entropy encoding of coeffs
  - N.B. DCCs not predicted in P/B frames
    → Already represent a differential value!
  - Each processed mblk transmitted along with motion vectors

NOTE1: Residuals very small
→ mblk may be '*skipped*' (values of ref mblk used)
→ decision made by encoder



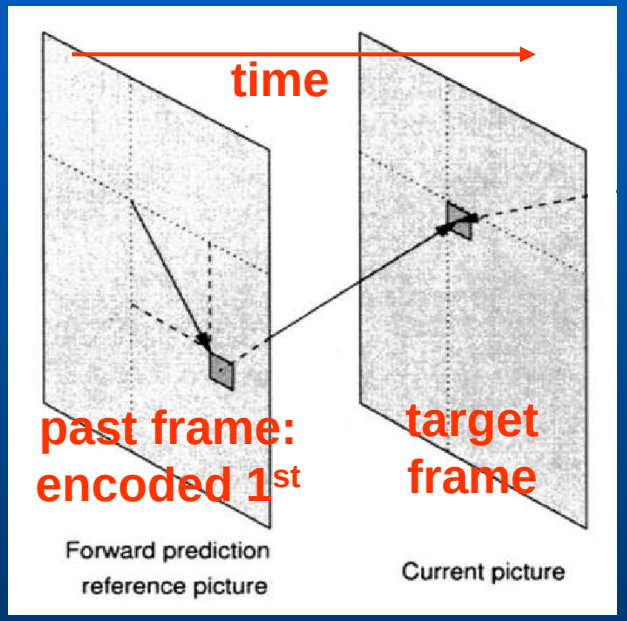
10.3.1 Inter Coding: P/B-frames

Encoding P-pictures (predicted from past I/P-pic)

– For each mblk in the *target* image…

- Motion Estimation
  → Ref image searched for best 16x16 match (→ motion vectors)
  → N.B. match does not have to align to mblk boundaries
- Target/reference mblks differenced
- Residual transformed by the 6 DCTs
- Quantization, RLE, Entropy encoding of coeffs
- N.B. DCCs not predicted in P/B frames
  → Already represent a differential value!
- Each processed mblk transmitted along with motion vectors

NOTE2: Residuals very large
→ mblk may be I-coded (i.e. no referencing)
→ decision made by encoder)



10.3.2 Inter Coding (cont.)

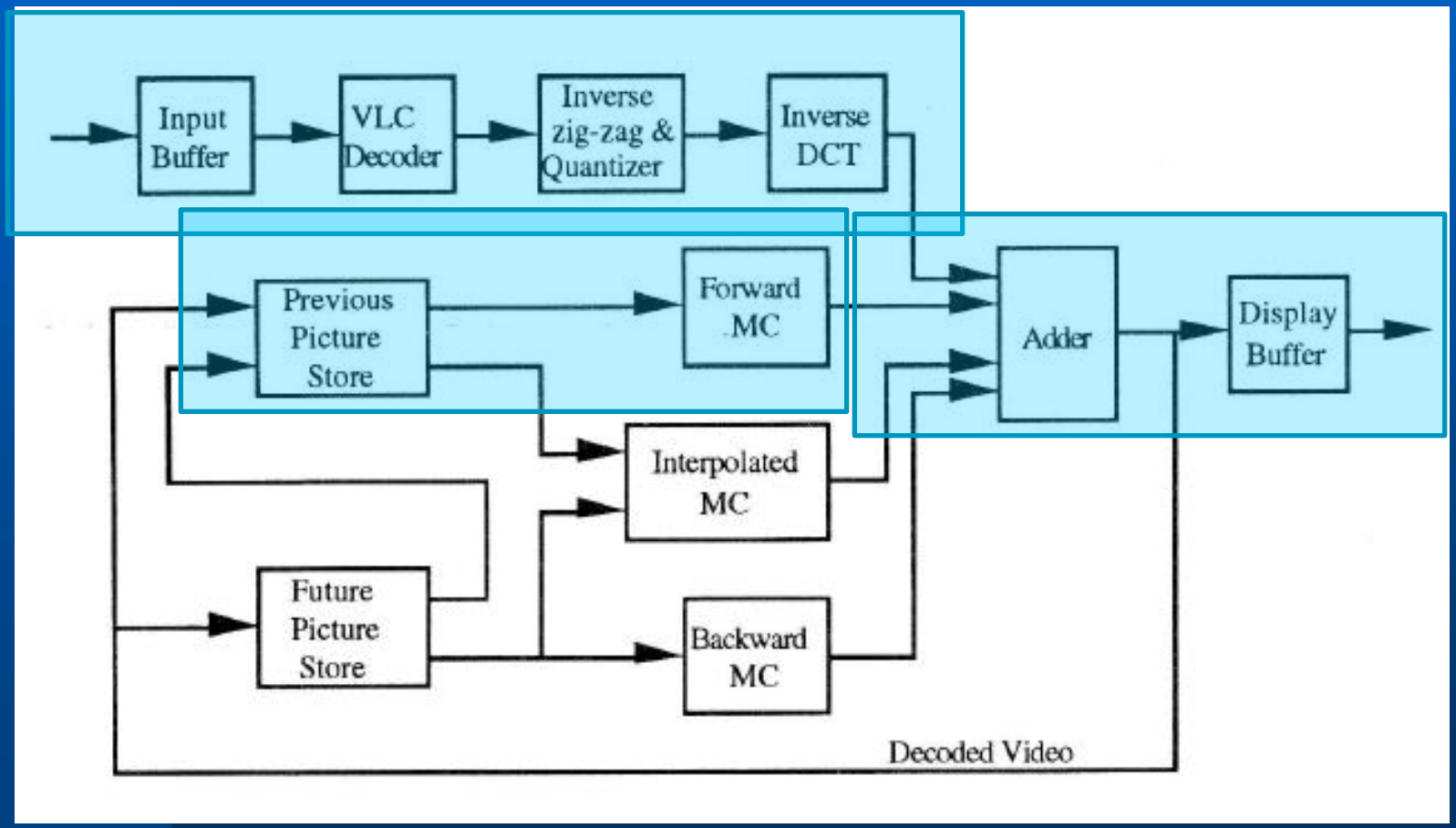
- **Decoding P-pictures**
  - Note, P-pics decoded with reference to a past I/P-pic already 'seen' by decoder
    - Decoder stores past pic for the referencing purpose
  - For each 16x16 mblk of *target* P-picture…
    - Entropy decoding, rescaling (QT), IDCT, etc.
      → Reconstructed prediction residuals (blocks: Y1-Y4, U, V)
    - MVs point to reference mblk (already reconstructed & stored)
      → Residuals + reference = reconstructed pixels

10.3.2 Inter Coding (cont.)

- **Decoding P-pictures**
  - Note, P-pics decoded with reference to a past I/P-pic already 'seen' by decoder
    - Decoder stores past pic for the referencing purpose
  - For each 16x16 mblk of *target* P-picture…
    - Entropy decoding, rescaling (QT), IDCT, etc.
      → Reconstructed prediction residuals (blocks: Y1-Y4, U, V)
    - MVs point to reference mblk (already reconstructed & stored)
      → Residuals + reference = reconstructed pixels

N.B. pixel values are in error due to lossy quantization in target AND reference image!

Decoding loop structure (P-picture)



10.3.2 Inter Coding (cont.)

- **Encoding B-pictures**
  - Similar to P-pictures, but *decision making* more complex…
  - For each mblk required to…
    - Search for the best forward (past) **AND** backward (future) MVs
    - Trade-off effectiveness Vs efficiency for prediction options…

10.3.2 Inter Coding (cont.)

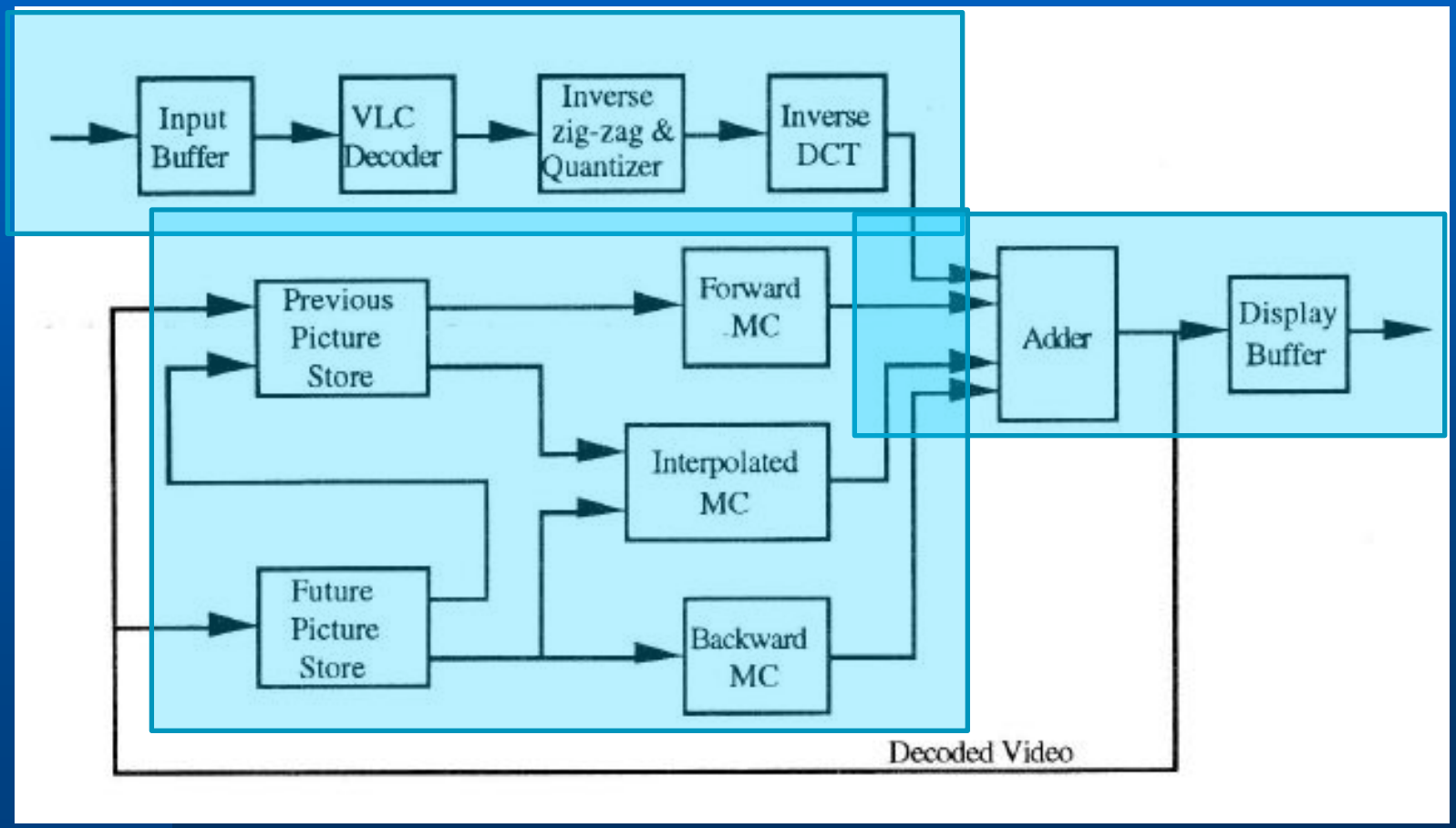
- **Encoding B-pictures**
  - Similar to P-pictures, but *decision making* more complex…
  - For each mblk required to…
    - Search for the best forward (past) **AND** backward (future) MVs
    - Trade-off effectiveness Vs efficiency for prediction options…
      - Is interpolated (bi-directional) motion compensation worthwhile?
      - Maybe just as effective to use forward OR backward prediction?
      - Maybe just as effective to intra-code?
      - Maybe mblk can be skipped (reference values used)?

10.3.2 Inter Coding (cont.)

- **Encoding B-pictures**
  - Similar to P-pictures, but *decision making* more complex…
  - For each mblk required to…
    - Search for the best forward (past) **AND** backward (future) MVs
    - Trade-off <u>effectiveness Vs efficiency</u> for prediction options…
      - Is interpolated (bi-directional) motion compensation worthwhile?
      - Maybe just as effective to use forward <u>OR</u> backward prediction?
      - Maybe just as effective to intra-code?
      - Maybe mblk can be skipped (reference values used)?

A simple average of the past & future reference mblks → Target mblk subtracted from average

Decoding loop structure (B-picture)



10.3.2 Summary

- **I-pictures**
  - Only i-mblks; DCCs predictively encoded; Used as reference
- **P-pictures**
  - Forward motion compensation from past ('seen') picture
    - p-mblks + forward MVs (some i-mblks, some skipped mblks)
  - No DCC prediction (Exception: i-mblk DCCs → may be predicted)
  - Used as reference
- **B-pictures**
  - Bi-directional (forward & backward) motion compensation
    - b-mblks + fwd/bkwrd MVs (some i/p-mblks, some skipped mblks)
  - No DCC prediction (exception: i-mblk--to--i-mblk)
  - Never used as reference

10.3.3 Quantization

QUANTIZATION

- As with JPEG: the key data reduction step in MPEG-1
- All MPEG-1 features (picture differencing, motion compensation, etc.) → designed to ensure as many DCT coeffs → 0 as possible!
- Note, **QTs differ for intra/inter-coded mblks**

Default QT for i-mblks

→ Designed to exploit insensitivity of HVS to higher-freqs

→ Finer step size for lower freqs

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

10.3.3 Quantization

QUANTIZATION

- As with JPEG: the key data reduction step in MPEG-1
- All MPEG-1 features (picture differencing, motion compensation, etc.) → designed to ensure as many DCT coeffs → 0 as possible!
- Note, **QTs differ for intra/inter-coded mblks**

QT for p/b-mblks

→ Flat table used since now dealing with predicted representations, NOT real pixel values

→ Hence, frequency interpretation invalidated (i.e. could have low-freq mblk with high valued high-freq residuals due to poor motion compensation!

16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16

(default step-size = 16)

10.3.3 Quantization (cont.)

- **Quantization scaling**
  - Quantization may be scaled on mblk-by-mblk basis
    → Greater flexibility; e.g. 'on-the-fly' bit rate control !
  - Implementation…
    - Bit stream provides space for '*Quantizer Scale Value*'
      - Multiplies into the QT matrix to give a new 'scaled' step-size for each coefficient
    - Gives rise to two intra (i)-mblk sub-types…
      - *Intra-d mblks* (specified by the VLC '1')
        - the current quantizer scale should be used
      - *Intra-q mblks* (specified by the VLC '01')
        - a further five bits (32 levels) are appended which specify a new quantizer scale

10.3.3 Quantization (cont.)

N.B. HVS quite sensitive to low freq variations
→ To maintain DCC accuracy
→ DCC step-sizes fixed (8)
→ **Scaling only applies to ACCs!**

- **Quantization scaling**
  - Quantization may be scaled on mblk-by-mblk basis
    → Greater flexibility; e.g. 'on-the-fly' bit rate control !
  - Implementation…
    - Bit stream provides space for '*Quantizer Scale Value*'
      - Multiplies into the QT matrix to give a new 'scaled' step-size for each coefficient
    - Gives rise to two intra (i)-mblk sub-types…
      - *Intra-d mblks* (specified by the VLC '1')
        - the current quantizer scale should be used
      - *Intra-q mblks* (specified by the VLC '01')
        - a further five bits (32 levels) are appended which specify a new quantizer scale

10.3.3 Quantization (cont.)

Quantization Rounding → actually differs per mblk type!

- **Intra-coded mblks (i-mblks)**
  - Coeff divided by corresponding step-size…
    - Result rounded to nearest integer
- **Inter-coded mblks (p/b-mblks)**
  - Coeff divided by corresponding step-size
    - Result rounded down (fractional part discarded)
      → Results in a wider interval around zero
      → More coeffs quantized to zero ☺
- N.B. Third type of rounding is sometimes used
  → Rounding towards minus infinity

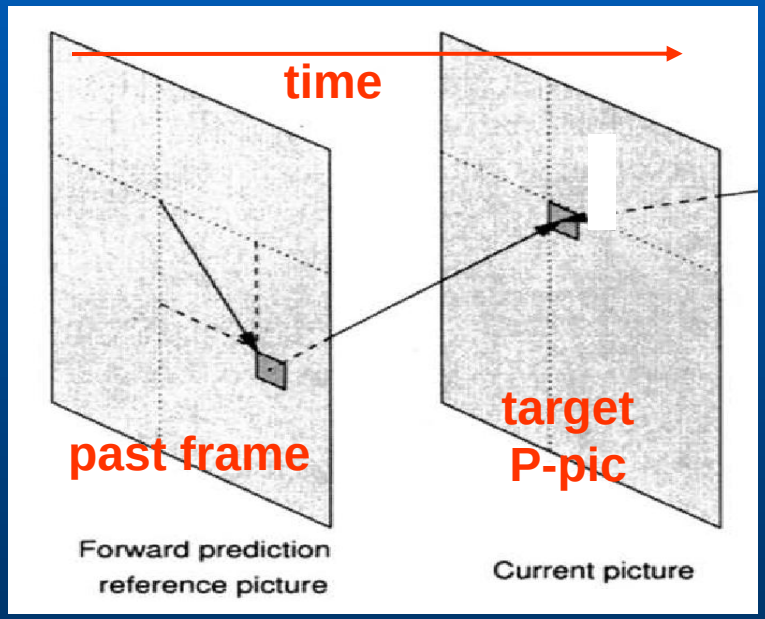
10.3.3 Quantization (cont.)

- **Quantization Rounding (Illustration)**

Operation	*Result*	*Operation*	*Result*	*Operation*	*Result*
Rounding to nearest integer		Rounding towards zero		Rounding towards $-\infty$	
4//4	1	4/4	1	4 DIV 4	1
3//4	1	3/4	0	3 DIV 4	0
2//4	1	2/4	0	2 DIV 4	0
1//4	0	1/4	0	1 DIV 4	0
(-1)//4	0	(-1)/4	0	(-1) DIV 4	-1
(-2)//4	-1	(-2)/4	0	(-2) DIV 4	-1
(-3)//4	-1	(-3)/4	0	(-3) DIV 4	-1
(-4)//4	-1	(-4)/4	-1	(-4) DIV 4	-1

10.3.4 Picture Types & Sequences

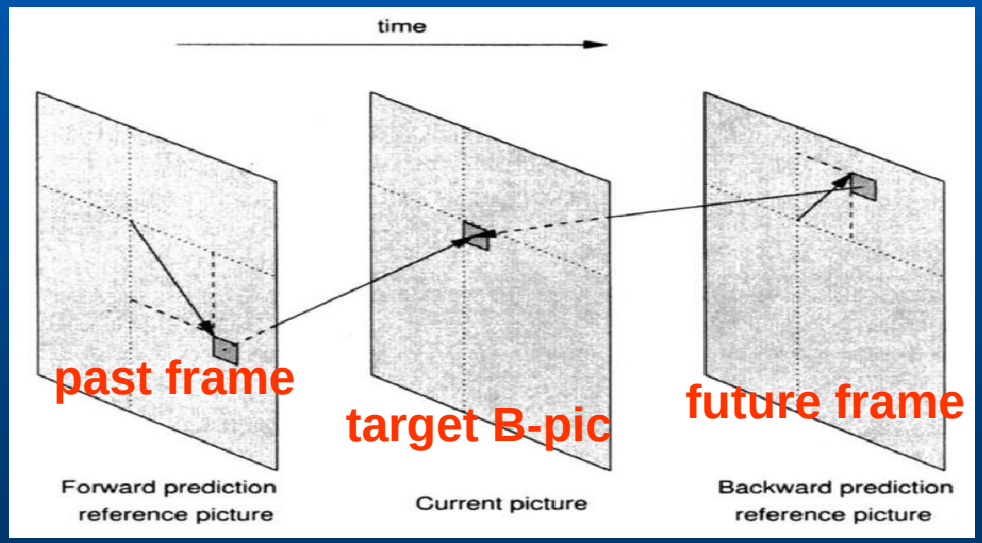
- **Recall: P-pictures**
  - Encoded with reference to past pic (already transmitted)



  - Decoded with reference to ‘seen’ pic (already decoded)

10.3.4 Picture Types & Sequences

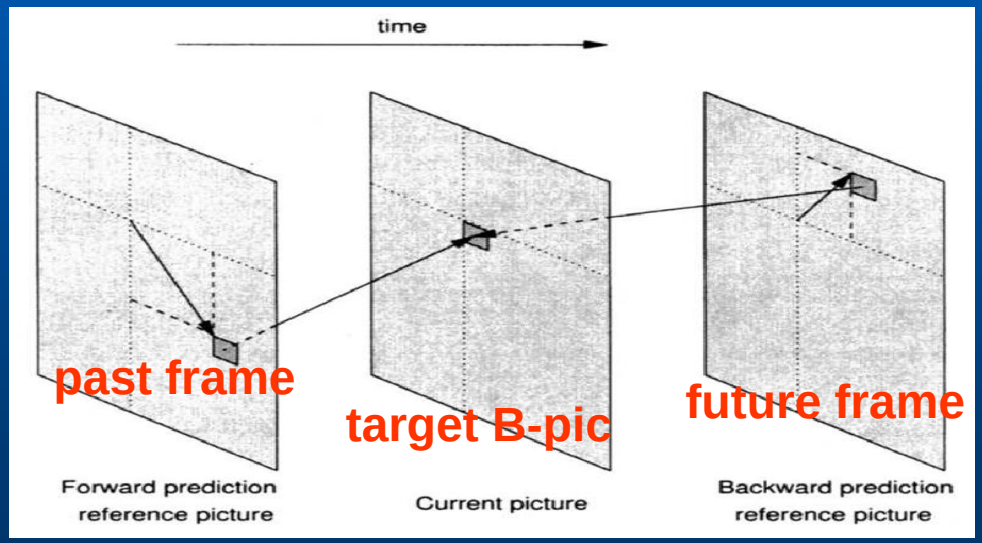
- **Recall: B-pictures**
  - Encoded with reference to past pic (already transmitted) and future pic (not yet seen)



  - Decoded with reference to ‘seen’ pic (already decoded) and future pic (not yet arrived)

10.3.4 Picture Types & Sequences

- **Recall: B-pictures**
  - Encoded with reference to past pic (already transmitted) and future pic (not yet seen)

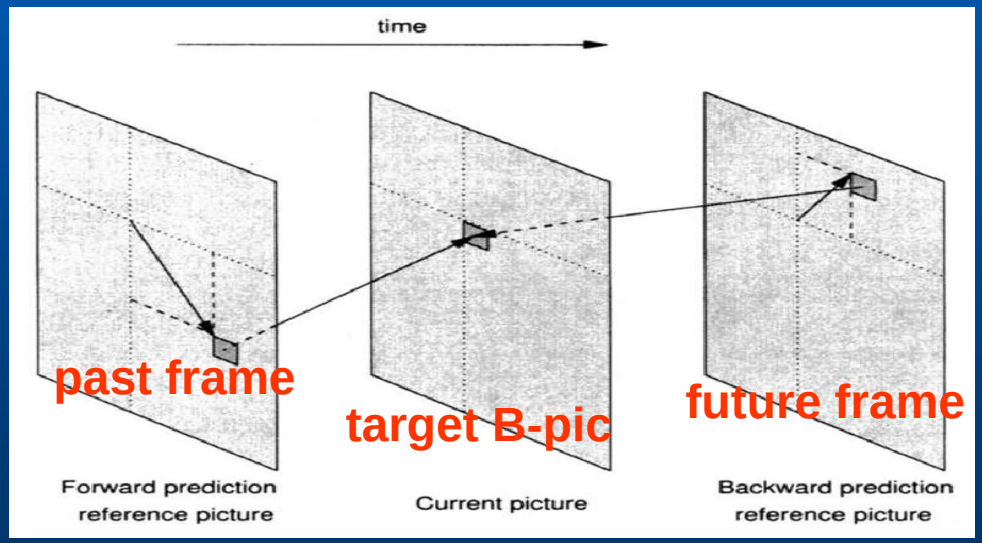


B-pic cannot be encoded until future pic is seen

  - Decoded with reference to 'seen' pic (already decoded) and future pic (not yet arrived)

10.3.4 Picture Types & Sequences

- **Recall: B-pictures**
  - Encoded with reference to past pic (already transmitted) and future pic (not yet seen)



B-pic cannot be encoded until future pic is seen

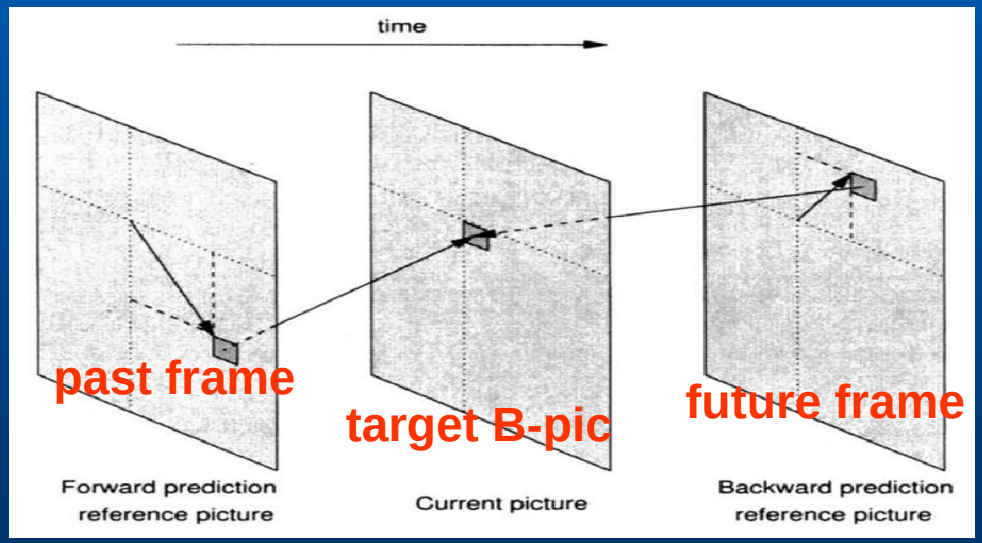
B-pic cannot be decoded until future pic has arrived and is decoded

  - Decoded with reference to 'seen' pic (already decoded) and future pic (not yet arrived)

10.3.4 Picture Types & Sequences

Makes sense to transmit future reference pics BEFORE dependant pics! (→ see GOPs…)

- **Recall: B-pictures**
  - Encoded with reference to past pic (already transmitted) and future pic (not yet seen)



B-pic cannot be encoded until future pic is seen

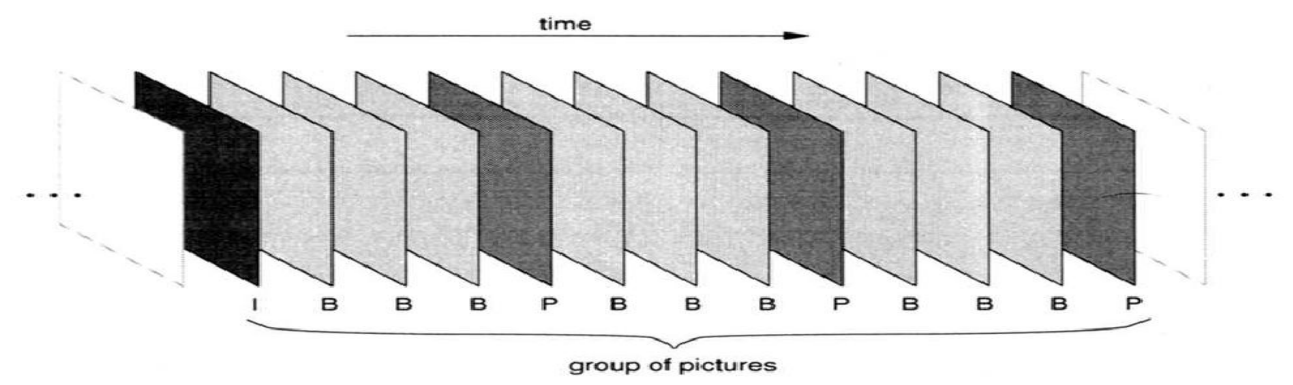
B-pic cannot be decoded until future pic has arrived and is decoded

  - Decoded with reference to 'seen' pic (already decoded) and future pic (not yet arrived)

10.3.4 Picture Types & Sequences (cont.)

Groups of Pictures (GOPs)

GOP = a collection of interdependent B/P-pics + an I-pic



- First picture = the I-pic (decoding starts afresh)
  - Hence → GOP length = distance between two I-pics

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing**
  - Typical length-15 GOP (frame numbers & picture types)…

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

  - '*Stream order*' → indicates order in which decoder recieves pics
    - i.e. order of images encoded in bitstream
  - '*Display order*' → indicates order in which pics should be displayed
    - i.e. order originally presented to encoder
  - Note they differ!
  - Next, explain how this works…

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

3. Next pic transmitted = 6th display order pic (P-coded)
→ predicted from the previous I-pic sent earlier

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

3. Next pic transmitted = 6th display order pic (P-coded)
→ predicted from the previous I-pic sent earlier

4. Next 2 pics transmitted = 4th & 5th display order pics (B-coded)
→ bi-directionally predicted from I-pic and P-pic already sent

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

3. Next pic transmitted = 6th display order pic (P-coded)
→ predicted from the previous I-pic sent earlier

4. Next 2 pics transmitted = 4th & 5th display order pics (B-coded)
→ bi-directionally predicted from I-pic and P-pic already sent

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

3. Next pic transmitted = 6th display order pic (P-coded)
→ predicted from the previous I-pic sent earlier

4. Next 2 pics transmitted = 4th & 5th display order pics (B-coded)
→ bi-directionally predicted from I-pic and P-pic already sent

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

3. Next pic transmitted = 6th display order pic (P-coded)
→ predicted from the previous I-pic sent earlier

4. Next 2 pics transmitted = 4th & 5th display order pics (B-coded)
→ bi-directionally predicted from I-pic and P-pic already sent

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

3. Next pic transmitted = 6th display order pic (P-coded)
→ predicted from the previous I-pic sent earlier

4. Next 2 pics transmitted = 4th & 5th display order pics (B-coded)
→ bi-directionally predicted from I-pic and P-pic already sent

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

3. Next pic transmitted = 6th display order pic (P-coded)
→ predicted from the previous I-pic sent earlier

4. Next 2 pics transmitted = 4th & 5th display order pics (B-coded)
→ bi-directionally predicted from I-pic and P-pic already sent

10.3.4 Picture Types & Sequences (cont.)

- **GOP Sequencing (Illustration)**
  - Typical length-15 GOP (frame numbers & picture types)…

2. Next 2 pics transmitted = 1st & 2nd display order pics (B-coded)
→ uni-directionally predicted from the previous I-pic sent earlier

1. First pic transmitted = 3rd display order pic (I-coded)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

3. Next pic transmitted = 6th display order pic (P-coded)
→ predicted from the previous I-pic sent earlier

4. Next 2 pics transmitted = 4th & 5th display order pics (B-coded)
→ bi-directionally predicted from I-pic and P-pic already sent

10.3.4 Picture Types & Sequences (cont.)

display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

- N.B. If GOP represents beginning of a section that has been cut (edited) from a larger video stream…
  - First two display order pics (B-encoded) may depend on I/P-pics from previous GOP no longer available
  - Editing program should set the '*broken-link*' flag to indicate they cannot be decoded

Terminology:

→ GOP where all pics predicted internally : '*Closed GOP*'

→ '*Open GOP*' : may contain pictures that reference I/P pics external to itself

10.3.4 Picture Types & Sequences (cont.)

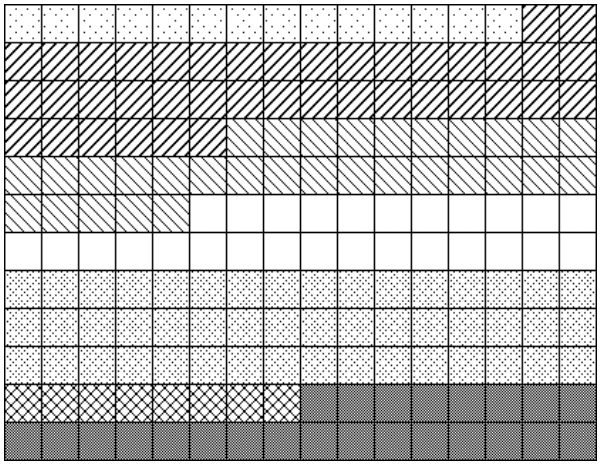
display order	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
picture Type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	P
stream order	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12

- **Final notes on GOPs…**
  - # B-pics between reference frames is variable
  - Can omit B-pics altogether : I/P-pics only!
    (Structure comparable to H.261)
  - Can also have an I-frame only sequence
    - i.e. GOP length of 1
    - Offers relatively low compression
    - Essentially equivalent to Motion JPEG

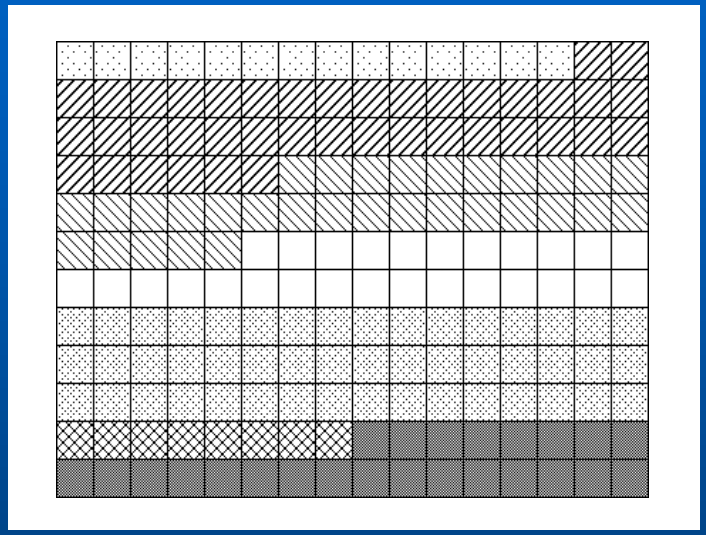
10.3.5 Sub-Picture Structures of MPEG-1

SLICES

- Slice = group of consecutive mblks
- Recall: DCCs of I-mblks predicted
  (Also, MVs predicted → see later)
- **Slices** → provide a way of controlling *error propagation* by reinitialising the DCC prediction
  - E.g. noisy channel → shorter slices should be used



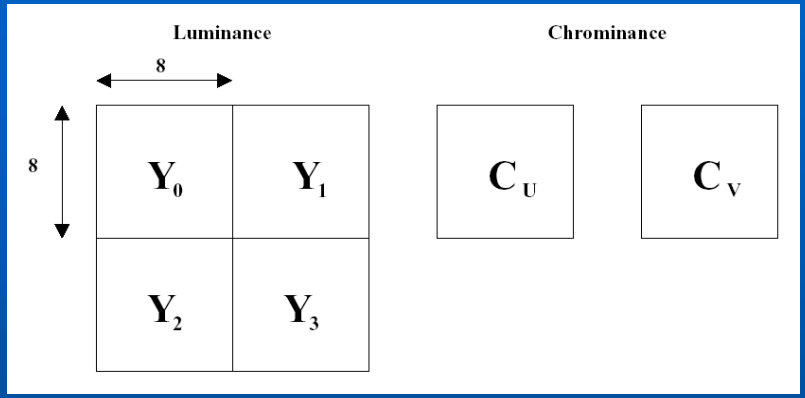
10.3.5 Sub-Picture Structures of MPEG-1



SLICES

- Slice = group of consecutive mblks
- Recall: DCCs of I-mblks predicted
  (Also, MVs predicted → see later)
- **Slices** → provide a way of controlling *error propagation* by reinitialising the DCC prediction
  - E.g. noisy channel → shorter slices should be used
- E.g. CIF pictures (352 x 288 pixels)…
  - A slice can consist of 1-396 mblks
  - Typical slice length: 22 mblks (→ 18 slices per frame)
- Slice header contains: Slice Start Code, Position code, Qzr scale
  - Header = sigificant overhead
    → trade-off between error prop. control and extra bits

10.3.5 Sub-Picture Structures of MPEG-1



MACROBLOCKS (mblks)

- 16x16 square of coloured pixels
  - 4X(8x8) Y + (8x8) U + (8x8) V
- Mblk header → Specifies mblk position within image
  - Address encoded as *relative* address, i.e. expressed as *increment* from last mblk sent

    → Clearly *Mblk_Address_Increment* always = 1 for I-pics (data transmitted for all mblks)
- Recall mblk types:
  - i-mblks (all pics)
  - p/b-mblks (P/B-pics)
  - skipped (P/B-pics)

Values taken as is from same mblk position in previous pic (no MVs or residuals coded) → Take up ZERO space in bitstream since presence *implied* by mblk address increment
E.g. increment = 3 → two mblks skipped!

10.3.5 Structures Within Pictures (cont.)

BLOCKS

- An 8x8 square array of monochrome pixels that can represent Y, U, or V pixel values
- N.B.
  - Blocks can be skipped within a mblk
    … and mblks may be skipped within a slice
  - However, the first block of any slice must be coded!

The encoder makes these decisions based on target bitrate specified

10.3.6 Motion Estimation

MOTION ESTIMATION (ME) in MPEG-1

- H.261: designed for low motion video conferencing
- MPEG-1: required to deal with high-motion video!
  - Hence → larger ME search range in MPEG-1

N.B. Assumed → Video motion may be sufficiently estimated from luminance information

- Hence → **ME search process only takes place in Y-domain**
  - Best Y-domain match determines reference mblks for Y, U & V pixel data

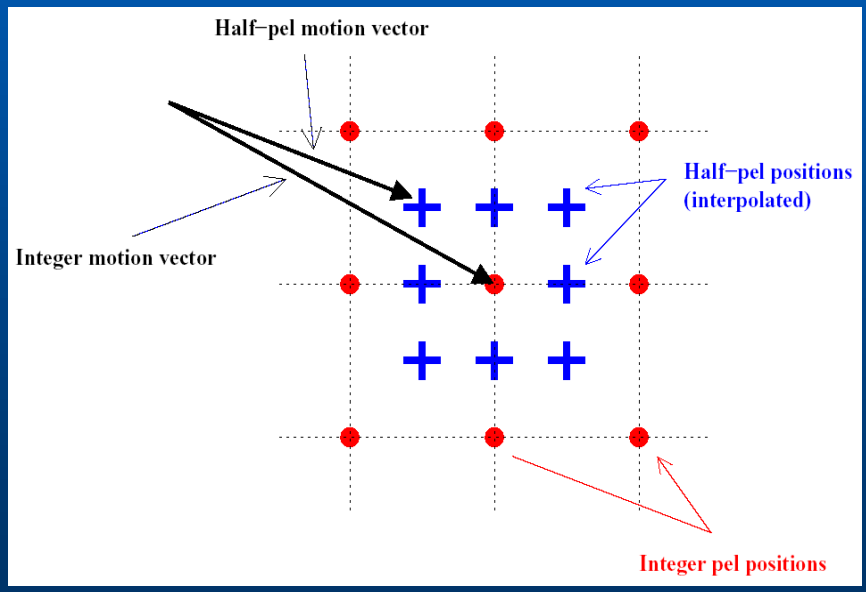
10.3.6 Motion Estimation (cont.)

SUB-PEL MOTION ESTIMATION

- MPEG-1 allows half-pel ME
- Why? → Moving objects not constrained by our pixel grids!

Implementation:

- Once the best pixel-level match found…
  - A further eight-position half-pel-level mblk search performed
    - Half-pel values formed by interpolation
    - Encoder decides whether half-pel referencing worthwhile



10.3.6 Motion Estimation (cont.)

- **ME Search Strategies**
  - MPEG-1 standard does NOT specify how the ME process should be performed at the encoder

10.3.6 Motion Estimation (cont.)

- **ME Search Strategies**
  - MPEG-1 standard does NOT specify how the ME process should be performed at the encoder

    E.g. Exhaustive Search
    - Simple, but computationally expensive…
    - Full search of all possible displacements
    - Guarantees finding minimum point in the matching function
    - Although optimal → not typically deployed since very computationally expensive

10.3.6 Motion Estimation (cont.)

- **ME Search Strategies**
  - MPEG-1 standard does NOT specify how the ME process should be performed at the encoder

    E.g. Exhaustive Search
    - Simple, but computationally expensive…
    - Full search of all possible displacements
    - Guarantees finding minimum point in the matching function
    - Although optimal → not typically deployed since very computationally expensive

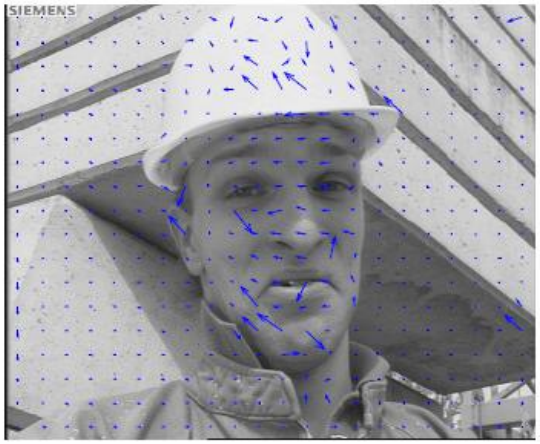
    Alternatives
  - Alternative strategies providing faster (albeit non-optimal) matching solutions
    - Logarithmic, Parallel One-Dimensional, etc. (other chapter)

10.3.6 Motion Estimation – motion vectors

Frame 66

Frame 69

Predicted frame69 with MV overlay



Predicted Frame69

10.3.6 *Coding of Motion Vectors*

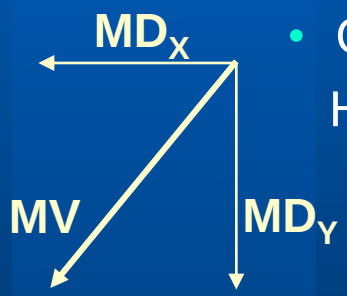
- **Consider: Video camera panning L→R …**
  - Scene moves R→L (new areas appearing on RHS, areas disappearing on LHS)
  - MVs from ME will be largely correlated in direction and magnitude
    - Coding independently → a redundant approach!

      Hence → MVs predictively (differentially) encoded!

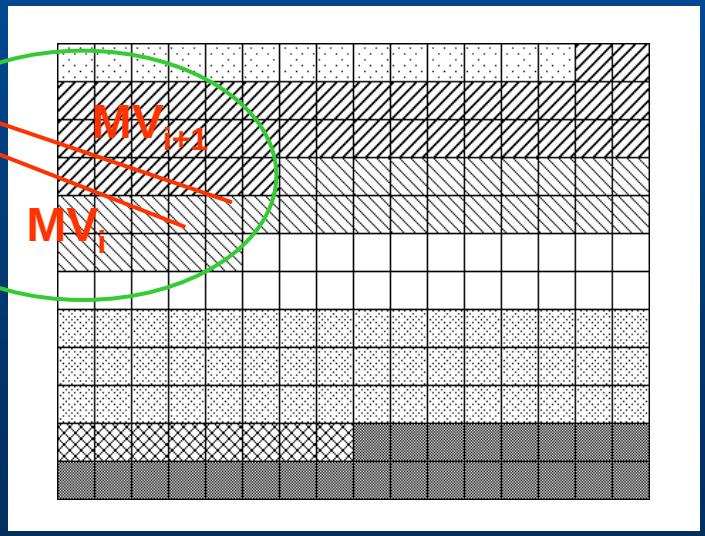
10.3.6 *Coding of Motion Vectors*

- **Consider: Video camera panning L→R …**
  - Scene moves R→L (new areas appearing on RHS, areas disappearing on LHS)
  - MVs from ME will be largely correlated in direction and magnitude
    - Coding independently → a redundant approach!

      Hence → MVs predictively (differentially) encoded!



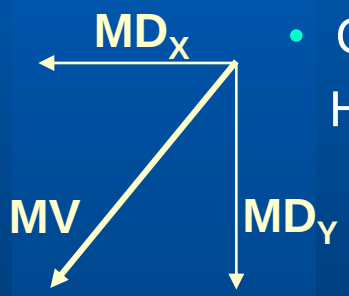
Subsequent *MD*s (X & Y motion displacements) subtracted yielding ***dMD***s (differential motion X & Y displacements)



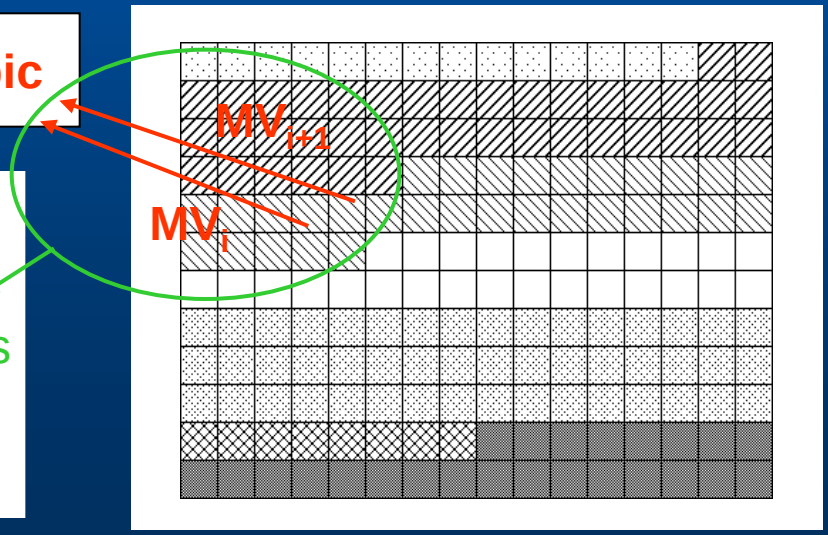
10.3.6 *Coding of Motion Vectors*

- **Consider: Video camera panning L→R …**
  - Scene moves R→L (new areas appearing on RHS, areas disappearing on LHS)
  - MVs from ME will be largely correlated in direction and magnitude
    - Coding independently → a redundant approach!

      Hence → MVs predictively (differentially) encoded!



Subsequent *MD*s (X & Y motion displacements) subtracted yielding ***dMD***s (differential motion X & Y displacements)



Prediction reinitialised…
(i) when new slice
(ii) after i-mblk
(iii) after skipped mblk

10.3.6 *Coding of Motion Vectors*

- **To obtain even more savings…**
  - The **max *dMD* size specified for each pic** (covers both x/y dirn)
    - Eliminates usage of needlessly long (bit) word-lengths in coding small *dMD* values
- **How? Best illustrated by example…**
  - First → the table of standard *dMD* size ranges in MPEG-1

10.3.6 *Coding of Motion Vectors*

- **To obtain even more savings…**
  - The **max *dMD* size specified for each pic** (covers both x/y dirn)
    - Eliminates usage of needlessly long (bit) word-lengths in coding small *dMD* values
- **How? Best illustrated by example…**
  - First → the table of standard *dMD* size ranges in MPEG-1

			Motion Vector Range $-16 \times f \leq dMD < 16 \times f$	
f_code	f	Modulus	full_pel=1	full_pel=0
1	1	32	-16 ... 15	-8 ... 7.5
2	2	64	-32 ... 31	-16 ... 15.5
3	4	128	-64 ... 63	-32 ... 31.5
4	8	256	-128 ... 127	-64 ... 63.5
5	16	512	-256 ... 255	-128 ... 127.5
6	32	1024	-512 ... 511	-256 ... 255.5
7	64	2048	-1024 ... 1023	-512 ... 511.5

10.3.6 *Coding of Motion Vectors*

- **To obtain even more savings…**
  - The **max *dMD* size specified for each pic** (covers both x/y dirn)
    - Eliminates usage of needlessly long (bit) word-lengths in coding small *dMD* values
- **How? Best illustrated by example…**
  - First → the table of standard *dMD* size ranges in MPEG-1

$dMD_{x/y} = MD_{x/y} - \text{prev}MD_{x/y}$

Indicates type of ME used in pic

			Motion Vector Range $-16 \times f \leq dMD < 16 \times f$	
f_code	f	Modulus	full_pel=1	full_pel=0
1	1	32	-16 ... 15	-8 ... 7.5
2	2	64	-32 ... 31	-16 ... 15.5
3	4	128	-64 ... 63	-32 ... 31.5
4	8	256	-128 ... 127	-64 ... 63.5
5	16	512	-256 ... 255	-128 ... 127.5
6	32	1024	-512 ... 511	-256 ... 255.5
7	64	2048	-1024 ... 1023	-512 ... 511.5

10.3.6 *Coding of Motion Vectors*

- **To obtain even more savings…**
  - The **max *dMD* size specified for each pic** (covers both x/y dirn)
    - Eliminates usage of needlessly long (bit) word-lengths in coding small *dMD* values
- **How? Best illustrated by example…**
  - First → the table of standard *dMD* size ranges in MPEG-1

$dMD_{x/y} = MD_{x/y}$ - prev$MD_{x/y}$

Indicates type of ME used in pic

			Motion Vector Range $-16 \times f \leq dMD < 16 \times f$	
f_code	f	Modulus	full_pel=1	full_pel=0
1	1	32	-16 ... 15	-8 ... 7.5
2	2	64	-32 ... 31	-16 ... 15.5
3	4	128	-64 ... 63	-32 ... 31.5
4	8	256	-128 ... 127	-64 ... 63.5
5	16	512	-256 ... 255	-128 ... 127.5
6	32	1024	-512 ... 511	-256 ... 255.5
7	64	2048	-1024 ... 1023	-512 ... 511.5

Each pic header contains 3-bit *f_code* that specifies a partic *dMD* range (differs for full-pel and half-pel cases)

10.3.6 *Coding of Motion Vectors*

- **To obtain even more savings…**
  - The **max *dMD* size specified for each pic** (covers both x/y dirn)
    - Eliminates usage of needlessly long (bit) word-lengths in coding small *dMD* values
- **How? Best illustrated by example…**
  - First → the table of standard *dMD* size ranges in MPEG-1

$dMD_{x/y} = MD_{x/y}$ - prev$MD_{x/y}$

Indicates type of ME used in pic

			Motion Vector Range $-16 \times f \leq dMD < 16 \times f$	
f_code	f	Modulus	full_pel=1	full_pel=0
1	1	32	-16 ... 15	-8 ... 7.5
2	2	64	-32 ... 31	-16 ... 15.5
3	4	128	-64 ... 63	-32 ... 31.5
4	8	256	-128 ... 127	-64 ... 63.5
5	16	512	-256 ... 255	-128 ... 127.5
6	32	1024	-512 ... 511	-256 ... 255.5
7	64	2048	-1024 ... 1023	-512 ... 511.5

f_code also specifies *f* and *Modulus values* → Used in processing *dMD* values to fit in range

Each pic header contains 3-bit *f_code* that specifies a partic *dMD* range (differs for full-pel and half-pel cases)

10.3.6 *Coding of Motion Vectors (cont.)*

- **E.g. Slice with following set of (e.g. x-dirn) MDs (*full_pel* = 1)…**

  3 10 30 30 -14 -16 27 24

  – Largest MD = 30

  – Assuming there are no larger MDs in the rest of the pic → $f = 2$

  – Differential (*dMD*) values are…

  3 7 20 0 -44 -2 43 -3 (i.e. new slice → initial prediction = 0)

			Motion Vector Range $-16 \times f \leq dMD < 16 \times f$	
`f_code`	f	Modulus	`full_pel=1`	`full_pel=0`
1	1	32	-16 ... 15	-8 ... 7.5
2	2	64	-32 ... 31	-16 ... 15.5
3	4	128	-64 ... 63	-32 ... 31.5
4	8	256	-128 ... 127	-64 ... 63.5
5	16	512	-256 ... 255	-128 ... 127.5
6	32	1024	-512 ... 511	-256 ... 255.5
7	64	2048	-1024 ... 1023	-512 ... 511.5

10.3.6 *Coding of Motion Vectors (cont.)*

- **E.g. Slice with following set of (e.g. x-dirn) MDs (*full_pel* = 1)…**

  3 10 30 30 -14 -16 27 24

  – Largest MD = 30

  – Assuming there are no larger MDs in the rest of the pic → $f = 2$

  – Differential (*dMD*) values are…

  3 7 20 0 -44 -2 43 -3 (i.e. new slice → initial prediction = 0)

  → N.B. Dynamic range now extended by one bit → undesirable!

			Motion Vector Range $-16 \times f \leq dMD < 16 \times f$	
`f_code`	f	Modulus	`full_pel=1`	`full_pel=0`
1	1	32	-16 ... 15	-8 ... 7.5
2	2	64	-32 ... 31	-16 ... 15.5
3	4	128	-64 ... 63	-32 ... 31.5
4	8	256	-128 ... 127	-64 ... 63.5
5	16	512	-256 ... 255	-128 ... 127.5
6	32	1024	-512 ... 511	-256 ... 255.5
7	64	2048	-1024 ... 1023	-512 ... 511.5

10.3.6 *Coding of Motion Vectors (cont.)*

			Motion Vector Range $-16 \times f \leq dMD < 16 \times f$	
f_code	f	Modulus	full_pel=1	full_pel=0
1	1	32	-16 ... 15	-8 ... 7.5
2	2	64	-32 ... 31	-16 ... 15.5
3	4	128	-64 ... 63	-32 ... 31.5
4	8	256	-128 ... 127	-64 ... 63.5
5	16	512	-256 ... 255	-128 ... 127.5
6	32	1024	-512 ... 511	-256 ... 255.5
7	64	2048	-1024 ... 1023	-512 ... 511.5

- **E.g. Slice with following set of (e.g. x-dirn) MDs (*full_pel* = 1)…**

  3 10 30 30 -14 -16 27 24

  – Largest MD = 30

  – Assuming there are no larger MDs in the rest of the pic → *f* = 2

  – Differential (*dMD*) values are…

  3 7 20 0 -44 -2 43 -3 (i.e. new slice → initial prediction = 0)

  → N.B. Dynamic range now extended by one bit → undesirable!

  – Process further using *Modulus*…

  - Where the dMDs exceed the allowed range {-32, 31} → add or subtract the corresponding *Modulus* value (64 for f=2) → this gives

  3 7 20 0 20 -2 -21 -3

  … values may now be coded on the basis of known upper bound, thus saving on bits! ☺

10.3.6 *Coding of Motion Vectors (cont.)*

- **Reversing the prediction process at the decoder:**

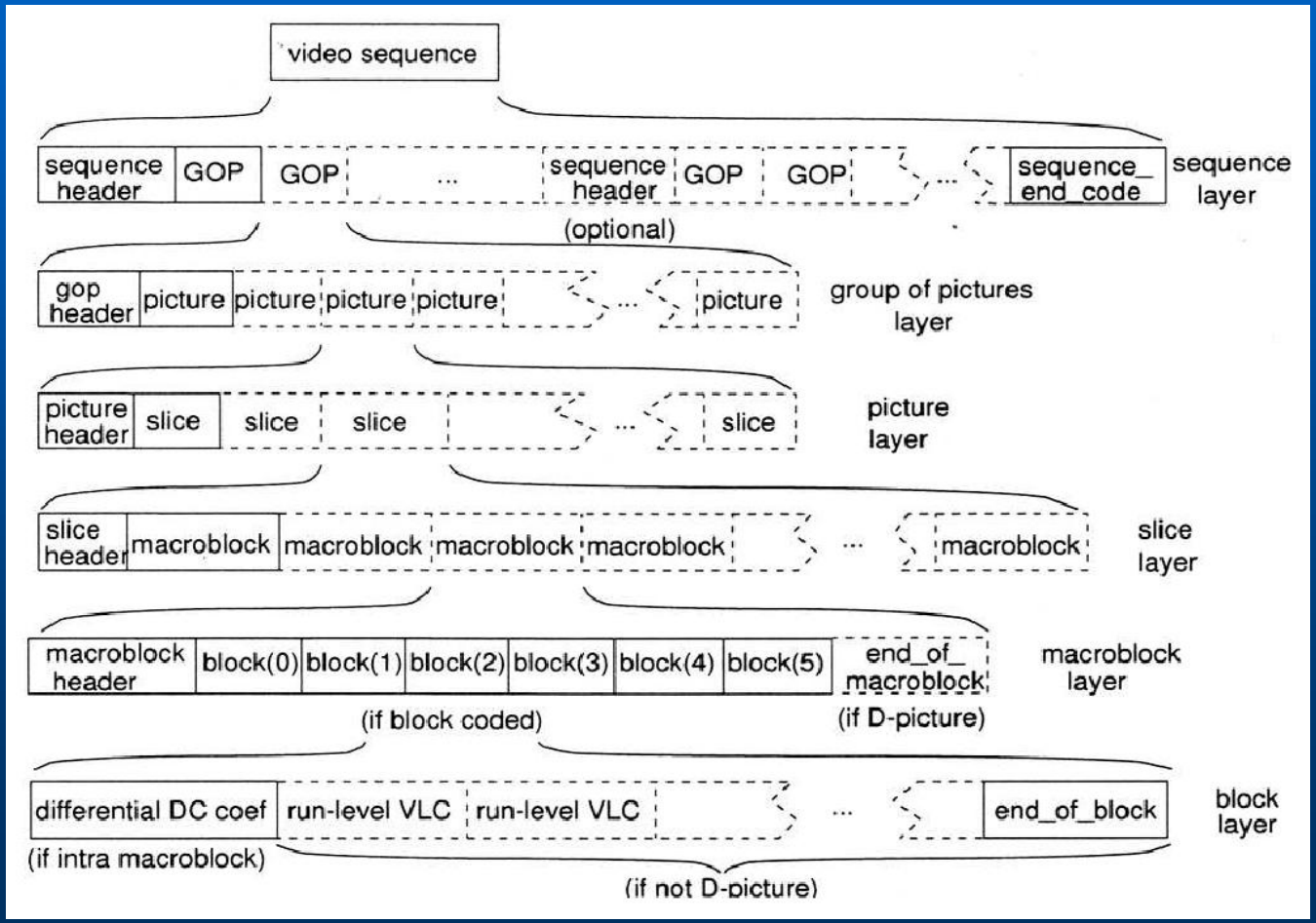
  3 7 20 0 20 -2 -21 -3

  - 3 + 7 = 10
  - 10 + 20 = 30
  - 30 + 0 = 30
  - 30 + 20 = 50 → 50 – 64 = -14
  - -14 – 2 = -16
  - -16 –21 = -37 → -37 + 64 = 27
  - 27 – 3 = 24

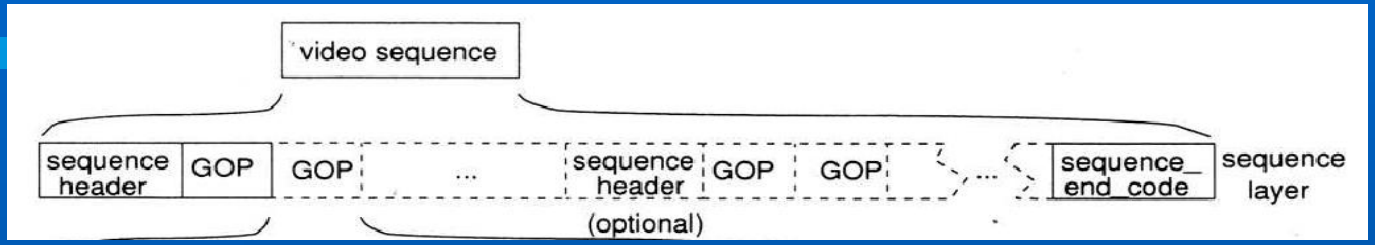
			Motion Vector Range $-16 \times f \leq dMD < 16 \times f$	
`f_code`	f	Modulus	`full_pel=1`	`full_pel=0`
1	1	32	-16 ... 15	-8 ... 7.5
2	2	64	-32 ... 31	-16 ... 15.5
3	4	128	-64 ... 63	-32 ... 31.5
4	8	256	-128 ... 127	-64 ... 63.5
5	16	512	-256 ... 255	-128 ... 127.5
6	32	1024	-512 ... 511	-256 ... 255.5
7	64	2048	-1024 ... 1023	-512 ... 511.5

10.3.7 Video Stream Layers

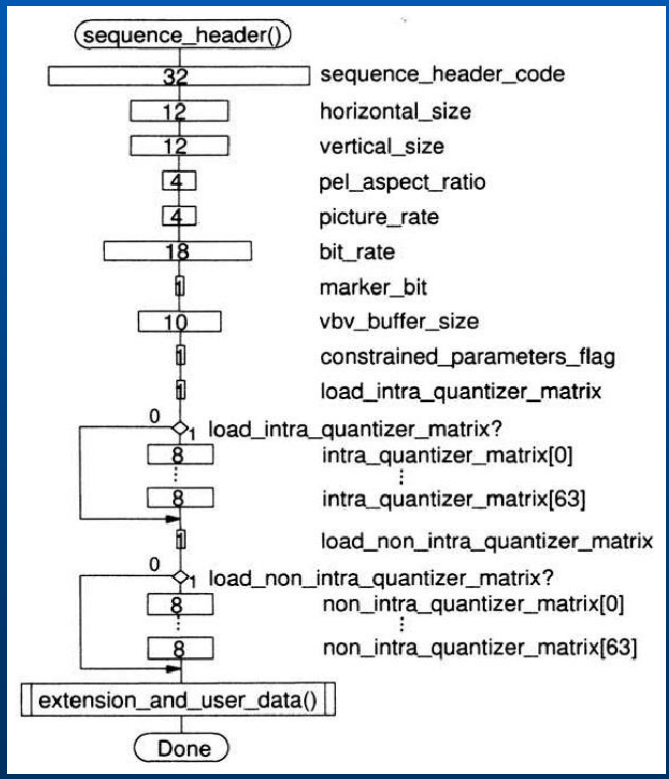
- **MPEG-1 video stream is hierarchically structured…**



10.3.7 Video Stream Layers (cont.)

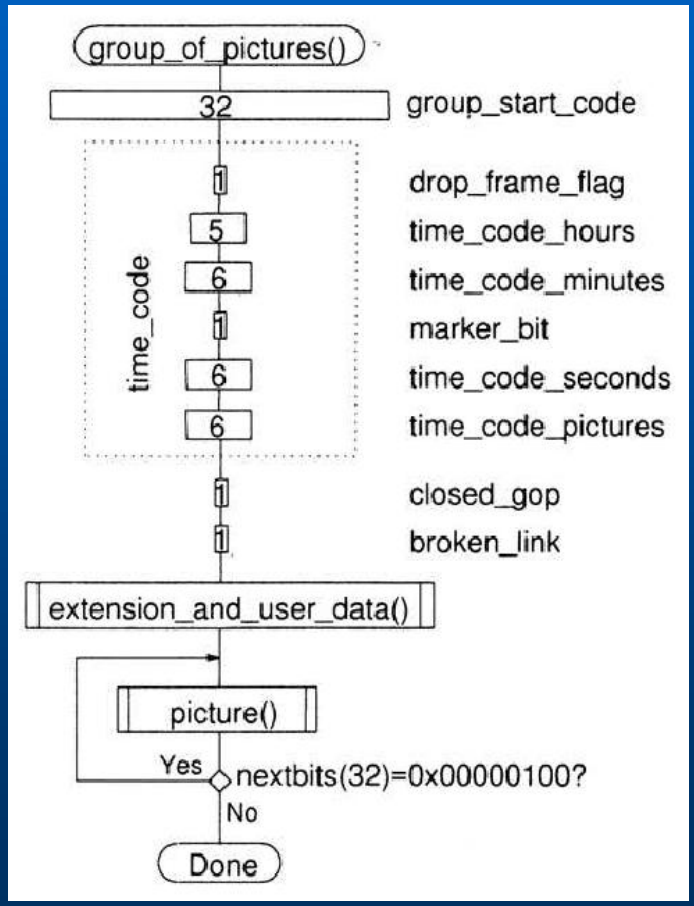


- **Sequence layer (contains GOPs)**
  - First element: Sequence Header
    - Must be at least one occurrence
    - May be repeated to facilitate random access or video editing
    - Not unusual to have a sequence header prior to each GOP
    - Most elements self explanatory
    - *bit_rate* and *vbv_buffer_size*
      - Inform decoder as to what memory allocation is sufficient to decode any pic combination occurring in the sequence



10.3.7 Video Stream Layers (cont.)

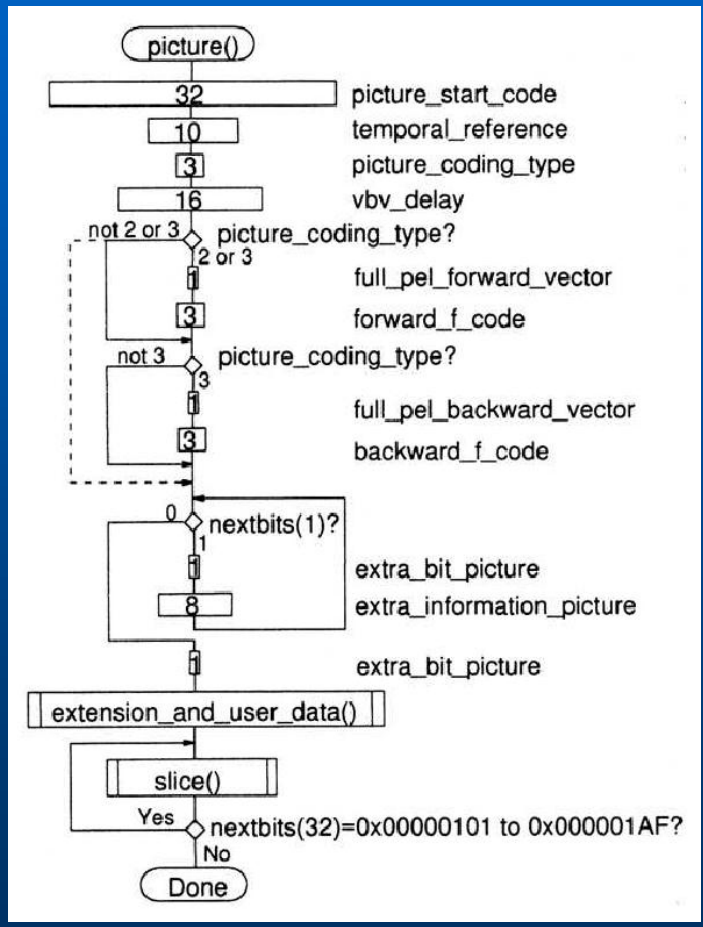
- **GOP layer (contains pictures)**
  - GOP header…
    - *drop_frame_flag*
      - Allows conversion from a 30Hz frame rate to 29.97Hz
        - by dropping frames at specific times
    - *time_code* values
      - Indicate time stamp associated with the first *displayed* picture
    - *closed_GOP* and *broken_link* flags
      - Relate to open/closed GOPs



10.3.7 Video Stream Layers (cont.)

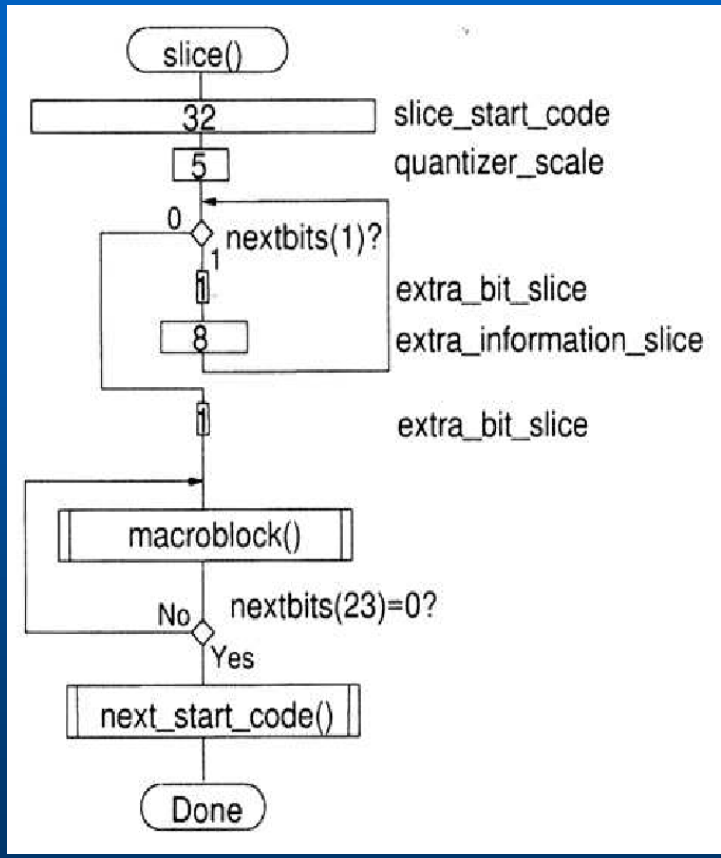
- **Picture layer (contains slices)**
  - Picture header…
    - *temporal_reference* flag
      - Indicates display order of pic within the sequence of pics following a sequence header
    - *picture_coding_type* flag
      - 1 : I-pic
      - 2 : P-pic
      - 3 : B-pic
      - 4 : D-pic
    - *f_code*: MV scale info for the picture (inter-coded pics only)

N.B. D-pic
→ Mode of MPEG where frames use DCCs only
→ Provides for rapid viewing (e.g. FF/FR)



10.3.7 Video Stream Layers (cont.)

- **Slice layer (contains mblks)**
  - Slice header…
    - *slice_start_code* flag
      - Contains a specification of the mblk row in which this slice starts
    - *quantiser_scale*
      - 5 bit element that can be adjusted from slice to slice
        - Key parameter for on-the-fly control over the bitrate and/or video quality



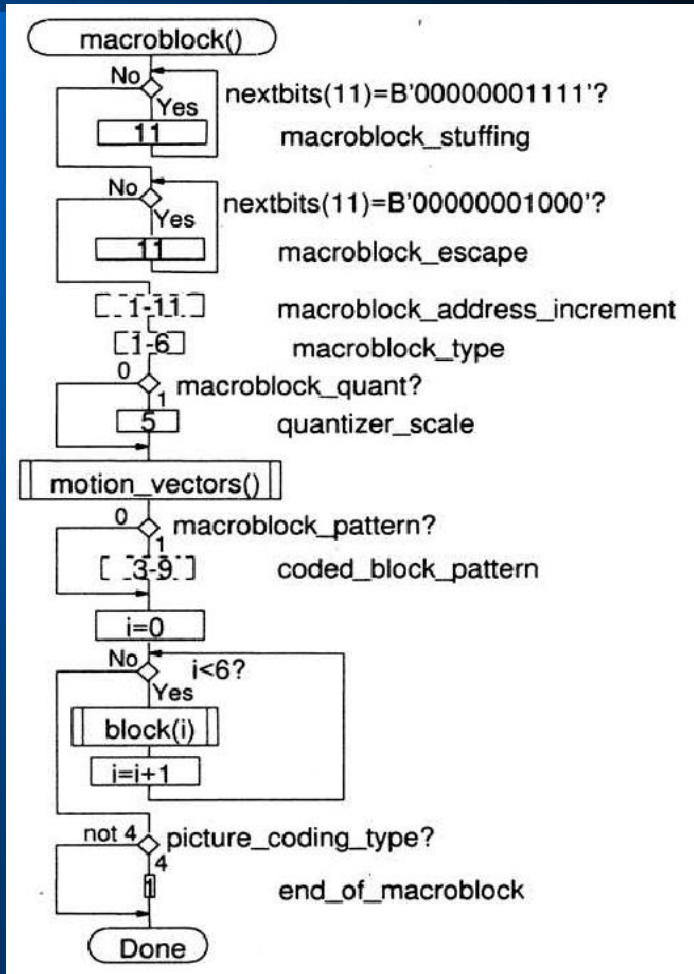
10.3.7 Video Stream Layers (cont.)

- **Macroblock layer (contains blocks)**
  - Macroblock header…
    - *macroblock_stuffing*
      - Fixed 11-bit pattern inserted into the stream when encoder needs to increase output bitrate
        - i.e. to maintain bitrate requirements of storage or transmission medium

Discarded by decoder

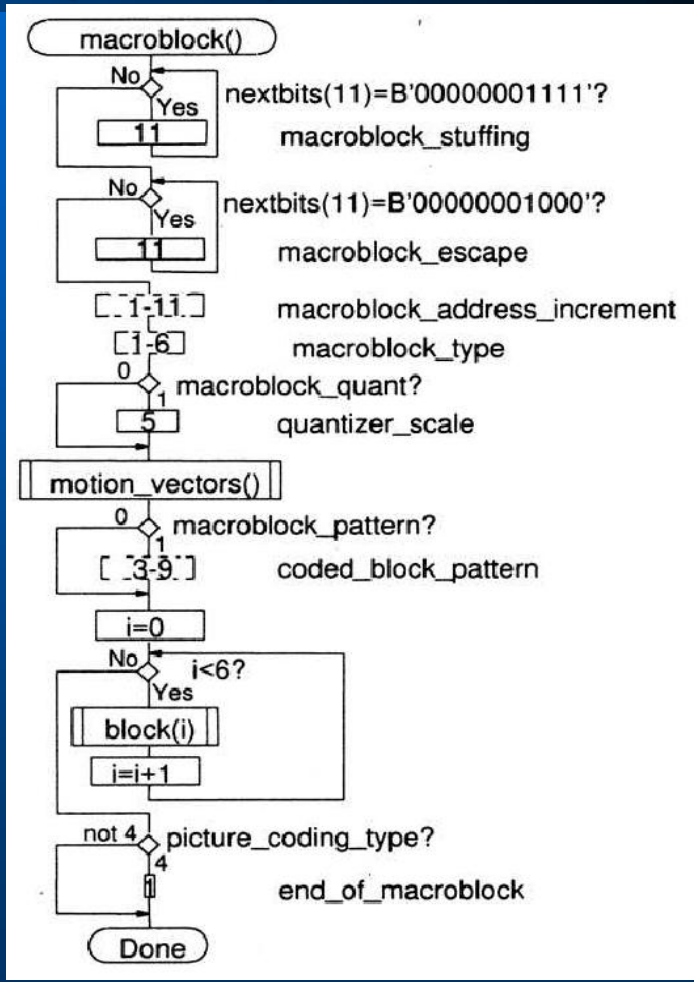
Eliminated in MPEG-2

    - *macroblock_address_increment*
      - VLC giving the difference btw address of this mblk and the previously transmitted mblk
      - Increments > 33 flagged by 11-bit *macroblock_escape* code
      - Increments > 66 indicated by two *macroblock_escape* codes



10.3.7 Video Stream Layers (cont.)

- **Macroblock layer (contains blocks)**
  - Macroblock header (cont.)
    - *previous_macroblock_address*
      - Absolute position of last non-skipped mblk
        - Set to default value at start of each slice
    - *quantiser_scale*
      - May reappear in the mblk header
        - Allows quantization to be adjusted at mblk level if necessary
    - *macroblock_type*
      - A VLC indicating method of coding (intra-d, intra-q, p, b, etc.)

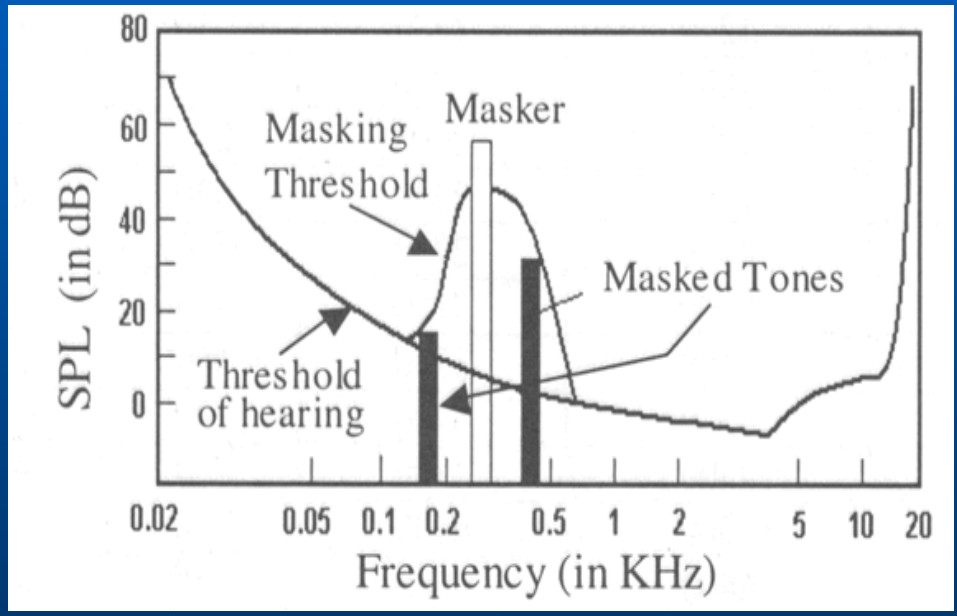


10.3.8 MPEG-1 Audio

- **Approach exploits *psychophysical* properties of human hearing, in particular…**
  - '**Masking effect**'
    - Presence of one audio signal (tone) '*masks*' perception of a smaller nearby signal

Quantize masked tones heavily and get away with it! ☺

- **MPEG audio**
  - →Takes advantage of our inability to hear quantization noise under such conditions



- **3 layers of audio encoding defined (increasing complexity)**
  - mp1, mp2, mp3 → see notes (10.3.8) on differences

10.3.9 MPEG-1 Systems Layer

- **Describes how the audio/video streams are combined in the overall stream**
  - Provides essential synchronization information to the decoder
    - Packets, presentation time stamps, system clock references, etc.
  - Allows each stream to be rendered and viewed in synch
  - Flexible
    - Up to 32 audio, 16 video and two data streams can be multiplexed together
  - Also responsible for buffer management…
    - Initialisation of buffering for playback start-up
    - Preventing buffer underflow
      - i.e. when decoder removes data too quickly from buffer
    - Preventing buffer overflow
      - i.e. when decoder does not remove data quickly enough

10.3.10 Video Formats & Picture Rates

- **Two most common analogue video standards…**
  - PAL (UK, British Commonwealth, most of Europe, China,…)
    - 625 lines per frame, 25fps (15.63kHz horizontal frequency) (digital PAL TV resolution – 720x576, interlaced)
  - NTSC (USA, Canada, Japan, Mexico,…)
    - 525 lines per frame, 30fps (15.75kHz horizontal frequency) (digital NTSC TV resolution – 720x480, interlaced)

- **Note, JPEG, MPEG, H.261, etc…**
  - Require image dimensions divisible by 8 (8x8 DCT encoding)
  - Furthermore → 16x16 motion compensation requires images divisible by 16
- **Commonly used resolution…**
  - CIF (Common Intermediate Format) → 352 x 288 pixels

10.3.11 The Constrained Parameter Flag

- **CPF: 1-bit flag contained in sequence header…**
  - Used to indicate if subsequent bitstream conforms to set of constraints that significantly reduce complexity of decoding required
  - Typical constraints…
    - *horizontal_size* $\leq$ 768
    - *vertical_size* $\leq$ 576
    - No. of mblks $\leq$ 396
    - *picture_rate* $\leq$ 30 per second
    - No. of mblks X *picture_rate* $\leq$ 396 X 25
    - *f_code* $\leq$ 4

10. MPEG-1: Video Compression

10.4 Other Uses of MPEG Bitstream

- **Main objective of MPEG-1 project…**
  - Compress video data
    - Meeting specific bandwidth requirements
    - Whilst maintaining acceptable quality
- **However…**
  - Encoded bitstream ends up being useful source of data from which we can infer *properties* about video content
    - i.e. for a range of non-compression purposes (e.g. IR)

E.g. can extract…

- DC coefficients
- Motion vectors
- Macroblock type
- Inter-frame dependency, etc.

10.4 Other Uses of MPEG Bitstream (cont.)

- **DC coeffs**
  - Can be used to infer low-fi statistics from the video
    - E.g. Coarse luminance/colour histograms of the pics

      → May be used as a fast and efficient basis for finding video periods that conform to a certain criterion (e.g. dark/bright periods)

- **Motion vectors…**
  - Really only mathematical block matching results
  - However → can still be used to infer magnitude and/or general direction of the motion between two pics
  - E.g. Identify periods of high action from a sequence!

10.4 Other Uses of MPEG Bitstream (cont.)

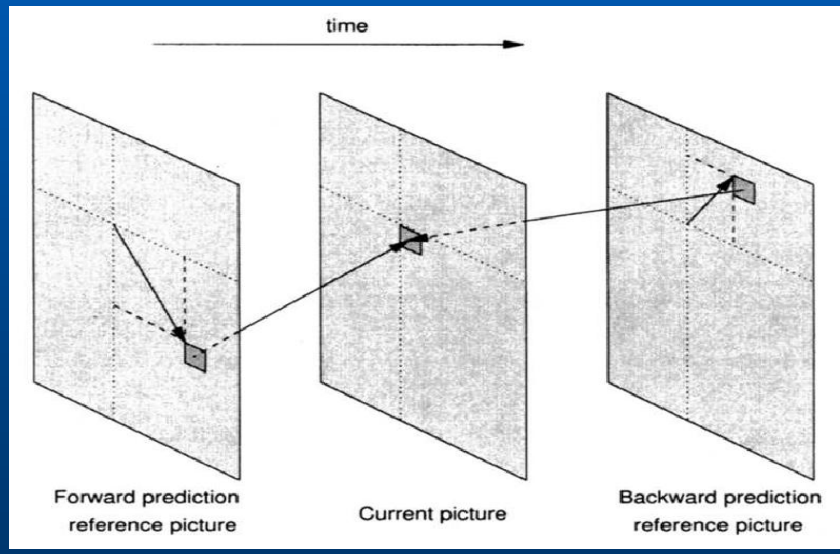
- **Note on video shots…**
  - Shot = Single unbroken piece of video from a single camera
    - Multiple shots → scenes
    - Multiple scenes → fully produced video
  - In world of video IR…
    - Shot = basic unit of video classification/retrieval

10.4 Other Uses of MPEG Bitstream (cont.)

- **Note on video shots…**
  - Shot = Single unbroken piece of video from a single camera
    - Multiple shots → scenes
    - Multiple scenes → fully produced video
  - In world of video IR…
    - Shot = basic unit of video classification/retrieval
  - Classic video analysis problem: **Shot boundary detection**
    - Shot transition types
      - Instantaneous 'hard cuts' (most frequently occurring)
      - Transitional, e.g. 'dissolves' and 'wipes'
      - Detecting hard cuts → particularly interesting problem in the context of MPEG-1!

10.4 Other Uses of MPEG Bitstream (cont.)

- **Detecting hard shot cuts in MPEG-1**
  - Can be done by exploiting inter-frame dependancy and mblk type
    → i.e., in a B-pic, we'd expect to see a mix of…
      - Some skipped mblks
      - Some I-mblks
      - Some P-mblks
      - Mostly B-mblks
    - Where this pattern disrupted
      - i.e. when most (if not all) B-pic references are to just one reference frame
        →Indicates that a very significant disruption has taken place
        → e.g. a hard shot cut



End of Chapter 10!