Video Compression

- INTRODUCTION TO VIDEO COMPRESSION
- VIDEO COMPRESSION WITH MOTION COMPENSATION
- SEARCH FOR MOTION VECTORS
- H.261

Recap: Why compression?

To reduce the volume of data to be transmitted.

- data types: text, fax, images, videos, audio …
- to reduce the bandwidth required for transmission
- to reduce the storage requirements.

Example: For standard definition TV,

Frame size	720*480 pixels
Frame rate	30 frames/second
Bits per pixel	24

Data rate: $30*(720*480)*24/8 = 31{,}104{,}000$ bytes/sec

For 2-hour SDTV clip, the required storage:

$$31{,}104{,}000*(60^2)*2 \cong 2.24*10^{11}\text{bytes}$$

Recap: Redundancy

Three types of data redundancy:

1. Coding redundancy
   - Unnecessarily assign a long code to a symbol. e.g. fixed-length coding

2. Spatial and temporal redundancy, e.g. pixel replication
   - Due to the spatial correlation between pixels in an *image*. e.g. a large area of one intensity.
   - Due to the temporal correlation between pixels in a *video*. e.g. between two consecutive frames.

3. Irrelevant information
   - Information that is ignored by Human Visual System. e.g. sounds that falls below the threshold of hearing.

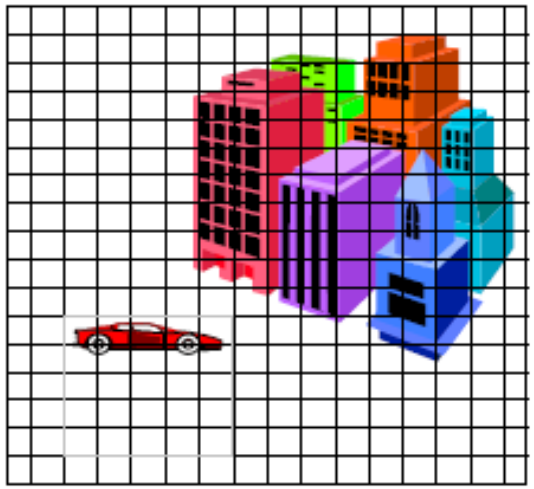
Video

A video consists of a time-ordered sequence of frames(images).

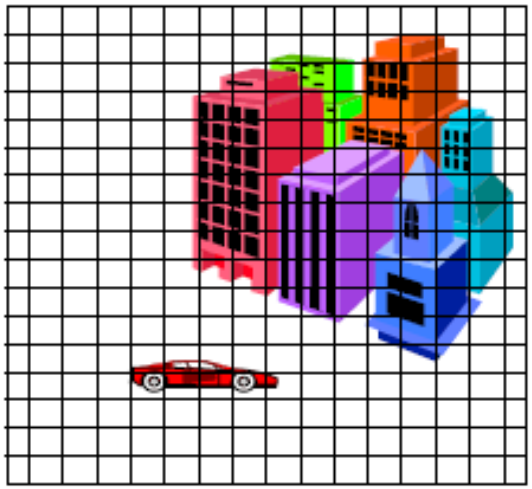
- The illusion of motion is given by displaying frames with a certain frequency. e.g. 30 frames/sec
- The number of frames showed each second depends on the spatial resolution of the frames (cinema, TV) as well as on the amplitude of the motion.

Video Compression

An obvious solution to video compression would be *predictive coding* based on previous frames.



Frame(N-1)



Frame(N)

Video Compression

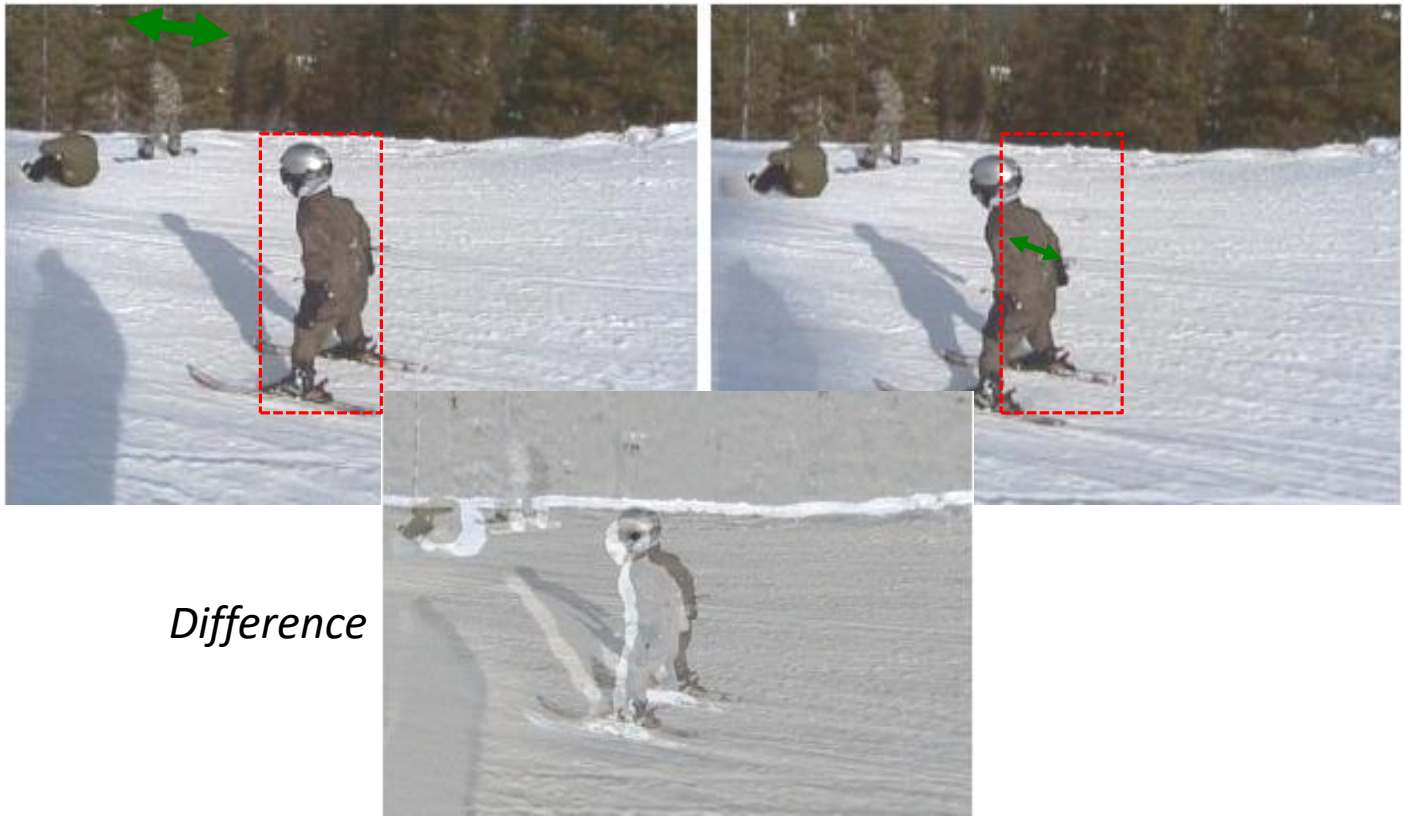
Compression proceeds by subtracting images: subtract in time order and code the residual error.

Frame(N-1)

Frame(N)

Difference (Residual)

Video Compression



Difference

It can be done even better by searching for just the right parts of the image to subtract from the previous frame.

Motion Compensation

Consecutive frames in a video are similar – so temporal redundancy exists.

Temporal redundancy is exploited so that not every frame of the video needs to be coded independently as a new image.

- The difference between the current frame and other frame(s) in the sequence will be coded - small values and low entropy,
- good for compression.

Motion Compensation

Algorithms has three main steps:

1. Motion estimation
   - Extracts motion info
   - Search for motion vectors

2. Motion compensation-based prediction
   - Reconstruct a predicted frame using MBs from reference frame(s) and their motion vectors.

3. Derivation of the prediction error
   - Record the difference between predicted frame and real frame

Motion Compensation

Each image is divided into macroblocks (MB) of size *NxN*.

- By default, *N* = 16 for luminance images. For chrominance images, *N* = 8 if 4:2:0 chrominance subsampling is adopted.

Motion compensation (MC) is performed at the MB level.

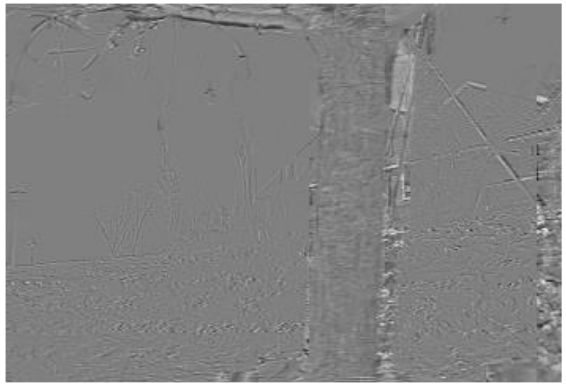
- The current image frame is referred to as Target Frame.
- A match is sought between the MB in the Target Frame and the most similar MB in previous and/or future frame(s) (referred to as Reference frame(s)).
- The displacement of the reference macroblock to the target macroblock is called a motion vector (MV) .

Motion-compensated prediction: example

Reference frame

Target frame

Target frame with displacement vectors



Motion-compensated Prediction error

Macroblock(MB)

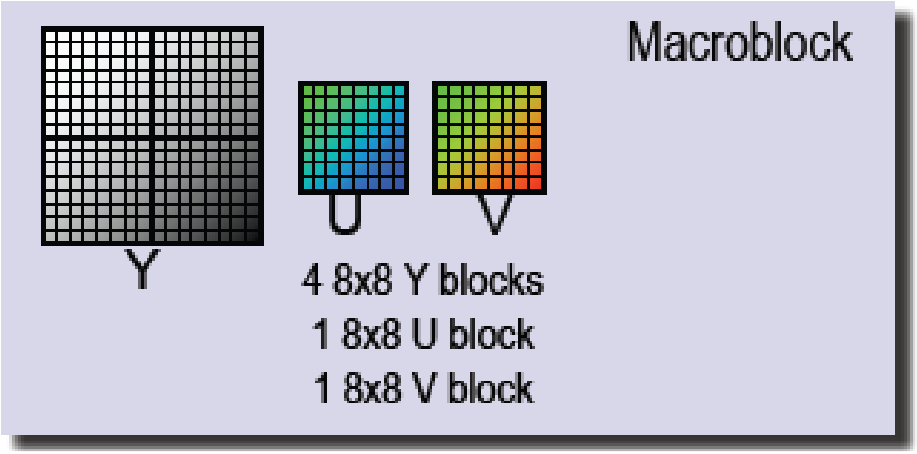
Recall:
What is the basic unit in JPEG compression?

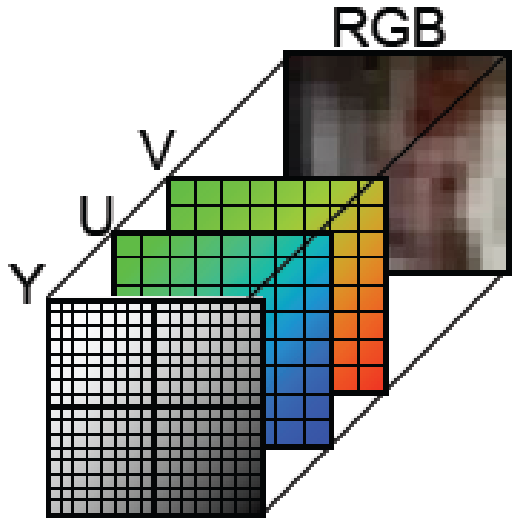
MB is basic unit for video compression.

Each MB is 16×16 pixels.

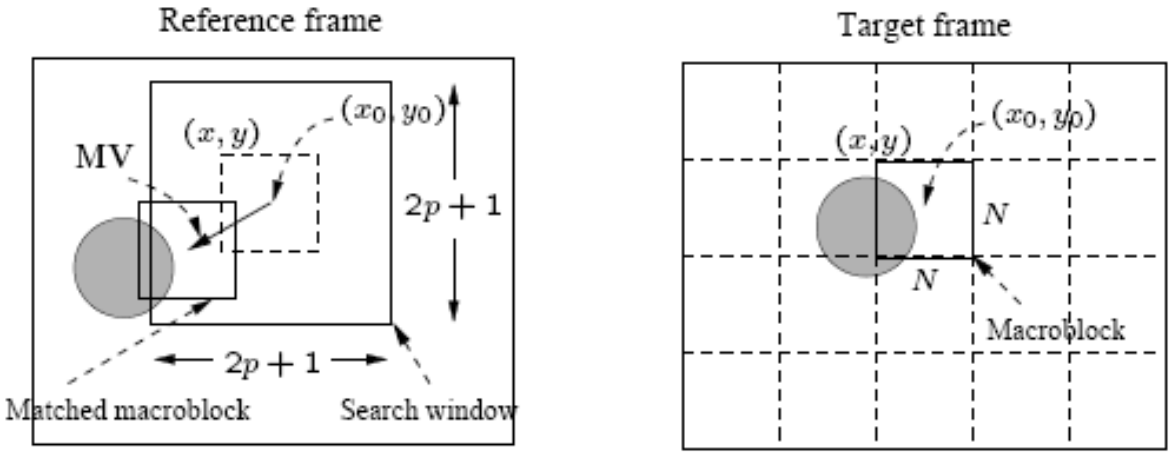
- Represent as YUV 4:2:0 data.
- 16×16 Luminance (Y) and subsampled 8*8 C_r , 8*8 C_b .

Represent this as 6 blocks of 8×8 pixels:





Search for Motion Vectors



MV search is computationally expensive.

MV search is usually limited to a small immediate neighborhood - both horizontal and vertical displacements in the range [$-p$, p]:

- This makes a search window of size ($2p$+1)x($2p$+1).

Search for Motion Vectors

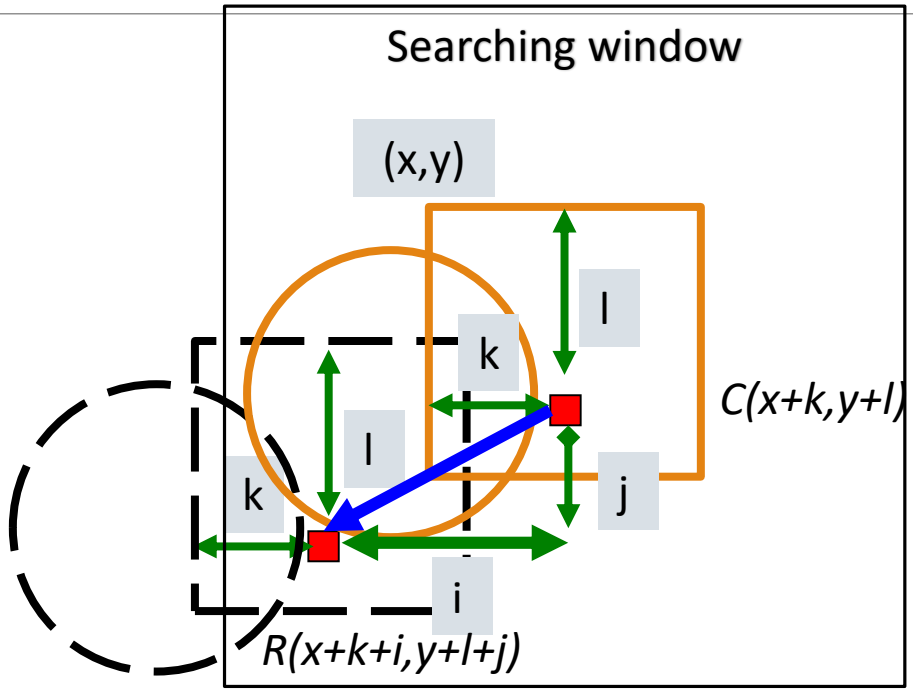
The difference between two MBs can be measured by their *Mean Absolute Difference (MAD)*:

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

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

The goal of the search is to find a vector (*i,j*) as the motion vector MV = (u,v), such that *MAD(i,j)* is minimum:

(*u,v*) = [(*i*, *j*) : *MAD*(*i,j*) is minimum, $i \in [-p, p]$, $j \in [-p; p]$]



Target frame

Sequential Search

Sequentially search the whole $(2p+1)$ *x* $(2p+1)$ window in the Reference frame (also referred to as Full search).

- A macroblock centered at each of the positions within the window is compared to the macroblock in the Target frame *pixel by pixel* and their respective *MAD* is then derived.
- The vector (i,j) that offers the *least MAD* is designated as the MV (u,v) for the macroblock in the Target frame.

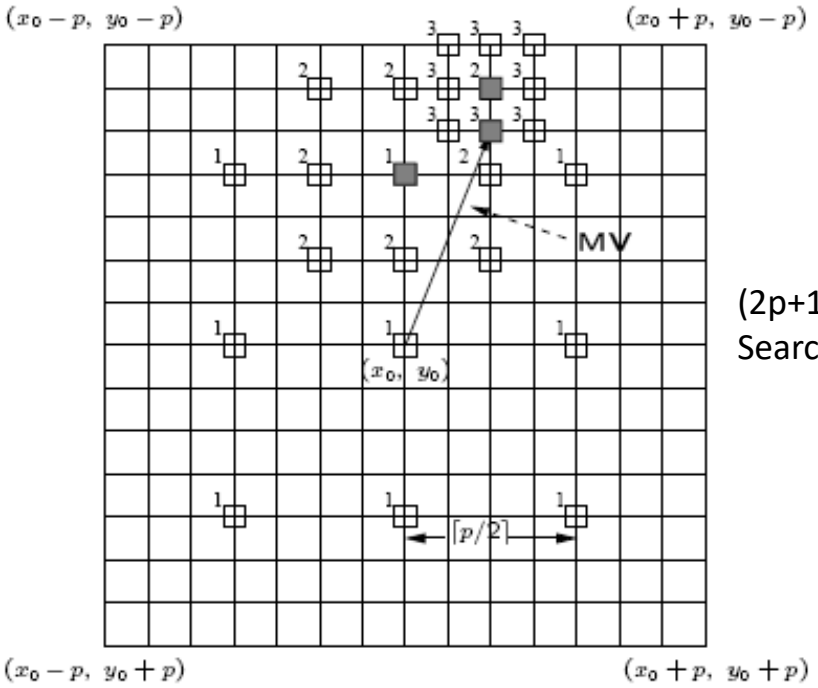
Sequential search method is very costly.

- assuming each pixel comparison requires three operations (subtraction, absolute value, addition), the cost for obtaining a motion vector for a single macroblock is $(2p+1).(2p+1).N^2 \cdot 3$ $)=> O(p^2N^2)$.

2D Logarithmic Search

- Logarithmic search: a cheaper version, that is suboptimal but still usually effective.
- The procedure for 2D Logarithmic Search of motion vectors takes several iterations and is similar to a binary search:

2D Logarithmic Search



(2p+1)*(2p+1)
Search window

1. Initially only nine locations in the search window are used as seeds for a MAD-based search; they are marked as `1'.

2. After the one that yields the minimum *MAD* is located, the center of the new search region is moved to it and the step-size ("offset") is reduced to half.

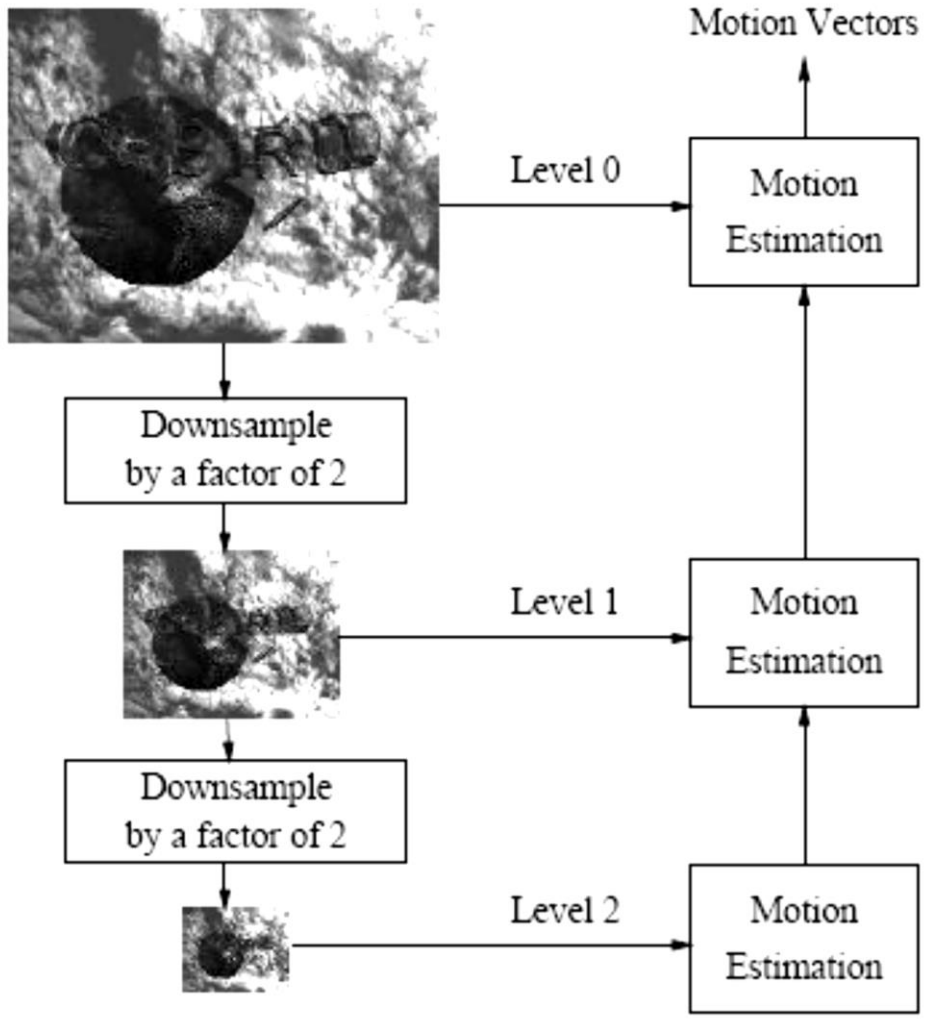
3. In the next iteration, the nine new locations are marked as `2', and so on.

Hierarchical Search

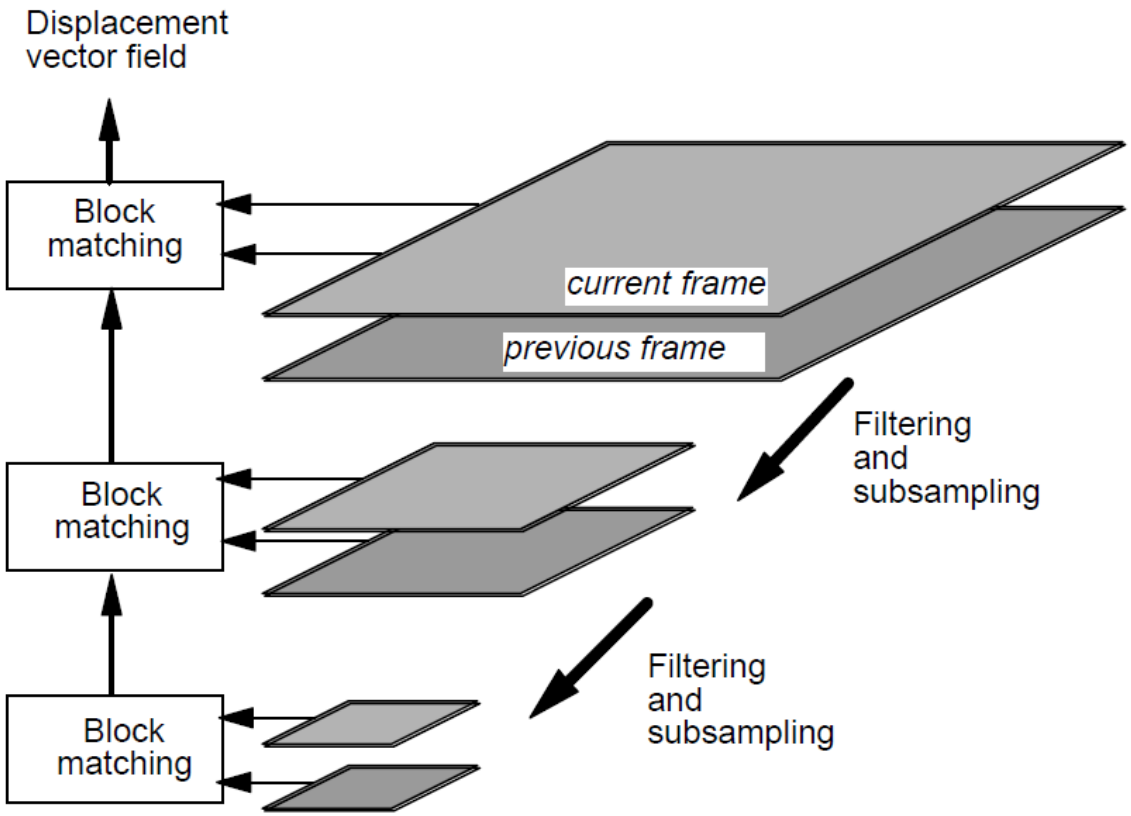
- The search can benefit from a hierarchical (multi-resolution) approach in which initial estimation of the motion vector can be obtained from images with a significantly reduced resolution.
- Next image: a three-level hierarchical search in which the original image is at Level 0, images at Levels 1 and 2 are obtained by down-sampling from the previous levels by a factor of 2, and the initial search is conducted at Level 2.
- Since the size of the macroblock is smaller and *p* can also be proportionally reduced, the number of operations required is greatly reduced.

Hierarchical Search

A Three-level Hierarchical Search for Motion Vectors



Hierarchical Search



Comparison: Motion Vector Search

Search Method	OPS_per_second for 720×480 at 30 fps	
	$p = 15$	$p = 7$
Sequential search	29.89×10^9	7.00×10^9
2D Logarithmic search	1.25×10^9	0.78×10^9
3-level Hierarchical search	0.51×10^9	0.40×10^9

Min cost!

Computational cost of motion vector search methods (unit: operations/second)

H.261

An earlier digital video compression standard, its principle of MC-based compression is retained in all later video compression standards.

The standard was designed for videophone, video conferencing and other audiovisual services over ISDN.

The video codec supports bit-rates of *px*64 kbps, where *p* ranges from 1 to 30 (Hence also known as *p** 64).

The delay of the video encoder be less than 150 msec so that the video can be used for real-time bi-directional video conferencing.

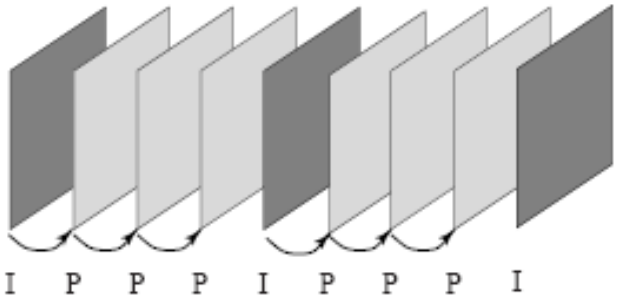
Video Formats Supported by H.261

Video format	Luminance image resolution	Chrominance image resolution	Bit-rate (Mbps) (if 30 fps and uncompressed)	H.261 support
QCIF	176 × 144	88 × 72	9.1	required
CIF	352 × 288	176 × 144	36.5	optional

CIF

QCIF

H.261 Frame Sequence



Two types of image: Intra-frames (I-frames) and Inter-frames (P-frames):

- I-frames are treated as independent images. Transform coding method similar to JPEG is applied within each I-frame, hence "Intra".
- To avoid propagation of coding errors, an I-frame is usually sent a couple of times in each second of the video.

H.261 Frame Sequence

- P-frames are not independent: coded by a forward predictive coding method (prediction from a previous P-frame is allowed - not just from a previous I-frame).

- Temporal redundancy removal is included in P-frame coding, whereas I-frame coding performs only spatial redundancy removal.

Motion vectors in H.261 are always measured in units of full pixel and they have a limited range of 15 pixels, i.e., $p = 15$.

H.261 structure

Video composed of frames

Each CIF frame composed of 12 Groups of Blocks (GOBs)

Each GOB is composed of 11x3 MacroBlocks

Each MB is 16x16 pixels

QCIF

CIF 352x288

GOB 0	GOB 1
GOB 2	GOB 3
GOB 4	GOB 5
GOB 6	GOB 7
GOB 8	GOB 9
GOB 10	GOB 11

QCIF 176x144

GOB 0
GOB 1
GOB 2

GOB and Resynchronization

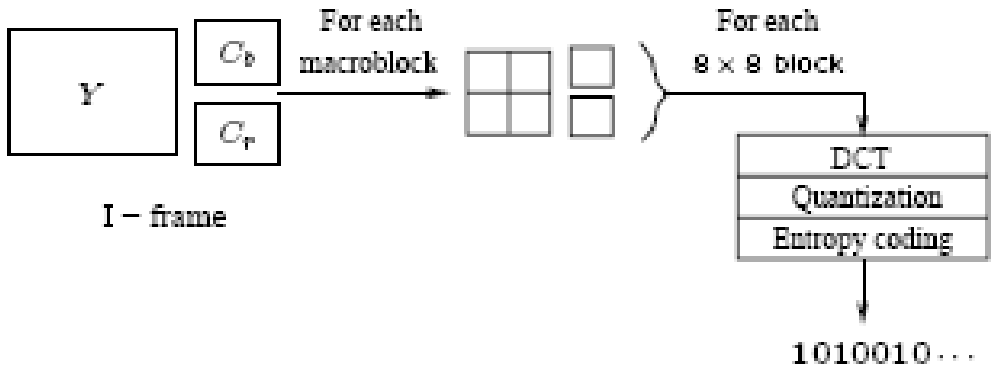
Purpose of Group of Blocks is resynchronization.

GOB starts with a sync code (binary: 00000000 00000001)

Within a GOB, encoded MBs don't even start on byte boundaries.

- If there's a bit error and you lose sync, or you join in the middle, you can't decode the next bits (you don't know where you are in the bit-stream).
- Scan for the next GOB sync code, and then you can start decoding.

Intra-frame (I-frame) Coding



Macroblocks are of size 16x16 pixels for the Y frame, and 8x8 for Cb and Cr frames, since 4:2:0 chroma subsampling is employed. A macroblock consists of four Y, one Cb, and one Cr 8x8 blocks.

For each 8x8 block a DCT transform is applied, the DCT coefficients then go through quantization zigzag scan and entropy coding.

Intra-frame (I-frame) Coding

Intra-coding of blocks is very similar to JPEG compression:

1. DCT
2. Quantization (Unlike JPEG, H.261 uses the same quantizer value for all coefficients.)
3. Zig-zag ordering.
4. Run-length encoding.
5. Huffman coding of what remains.

Inter-frame (P-frame) Predictive Coding

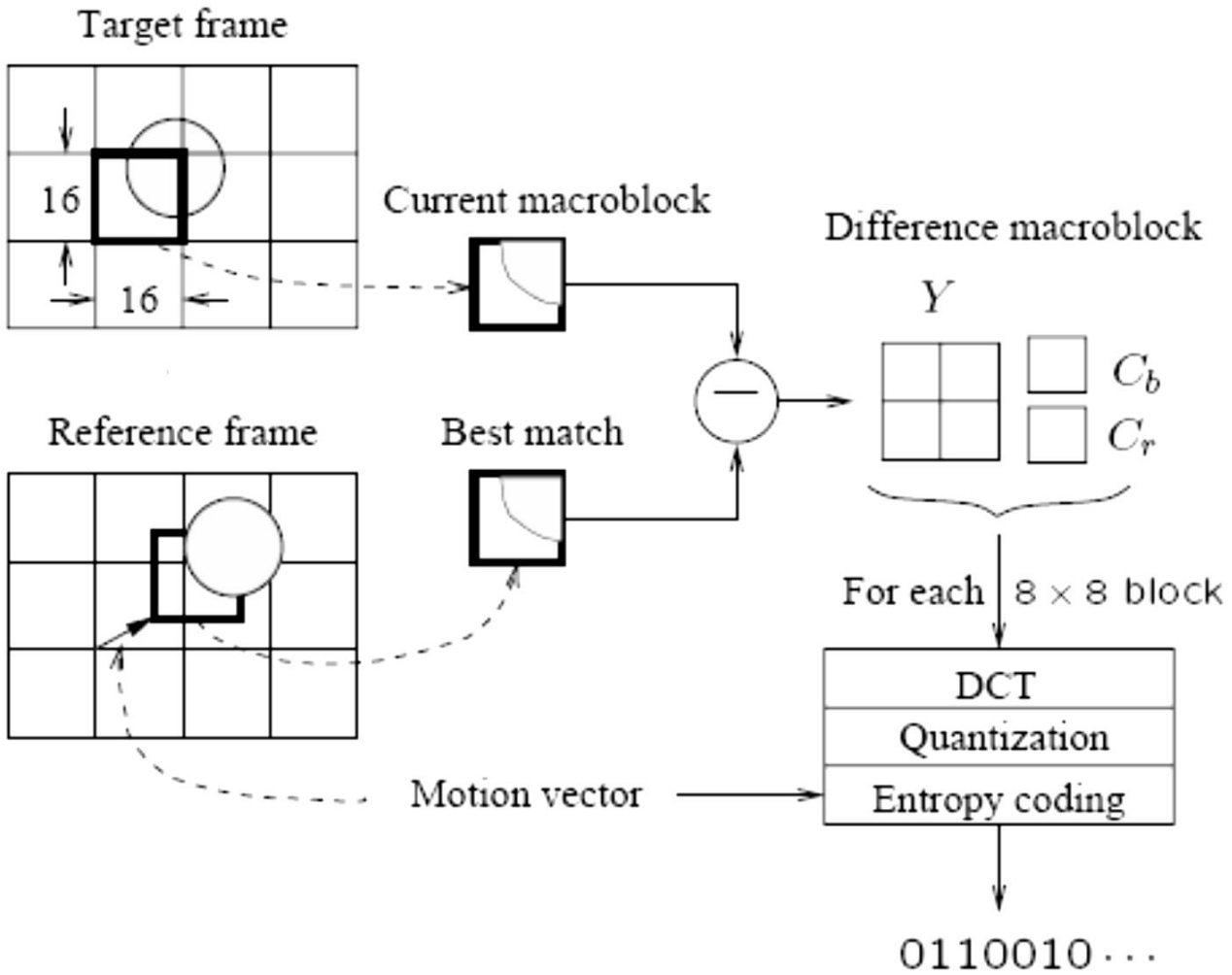
- For each macroblock in the Target frame, a motion vector is allocated by one of the search methods discussed earlier (e.g. full search).
- After the prediction, a difference macroblock is derived to measure the prediction error.
- Each of these 8x8 blocks go through DCT, quantization, zigzag scan and entropy coding procedures.

Inter-frame (P-frame) Predictive Coding

- The P-frame coding encodes the difference macroblock (not the Target macroblock itself).
- Sometimes, a good match cannot be found, i.e., the prediction error exceeds a certain acceptable level.
  - - The MB itself is then encoded (treated as an Intra MB)
  - and in this case it is termed a *non-motion compensated*
  - *MB*.
- For motion vector, the difference MVD is sent for entropy coding:

$$MVD = MV_{Preceding} - MV_{Current}$$

Motion Compensation Based P-frame Coding



Frame Differencing

Often the amount of information in the difference between two frames is a lot less than in the frame itself.

Frame 1

Frame 2

Difference

Motion

Motion in the scene will increase the differences.

If you can figure out the motion (where each block came from in the previous frame):

- Encode the motion as a motion vector (two small integers indicating motion in x and y directions)

- Encode the differences from the moved block using DCT + quantization + RLE + Huffman encoding.

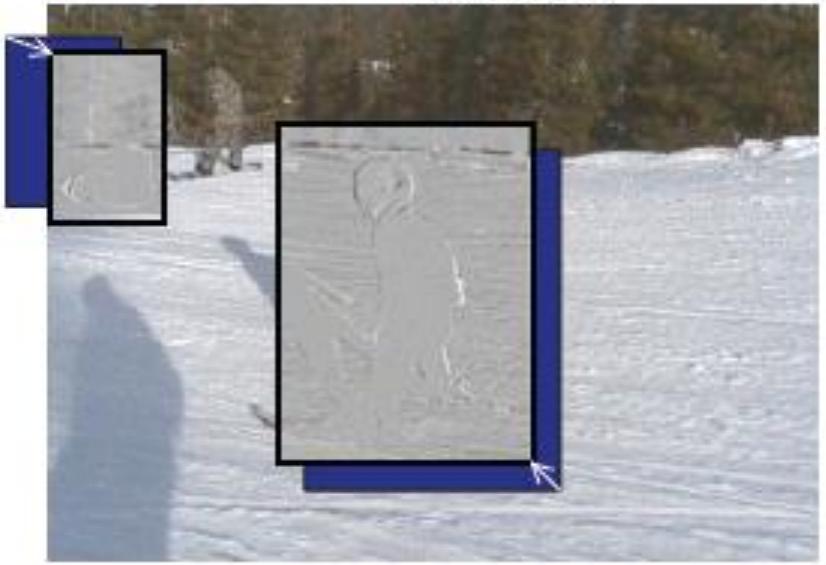
Motion

Frame 1

Frame 2

Frame 2 - 1
(lots of motion)

Coding from moved part of previous image can reduce the differences



Quantization in H.261

The quantization in H.261 uses a constant step size, for all DCT coefficients within a macroblock.

for DC coefficients in Intra mode:

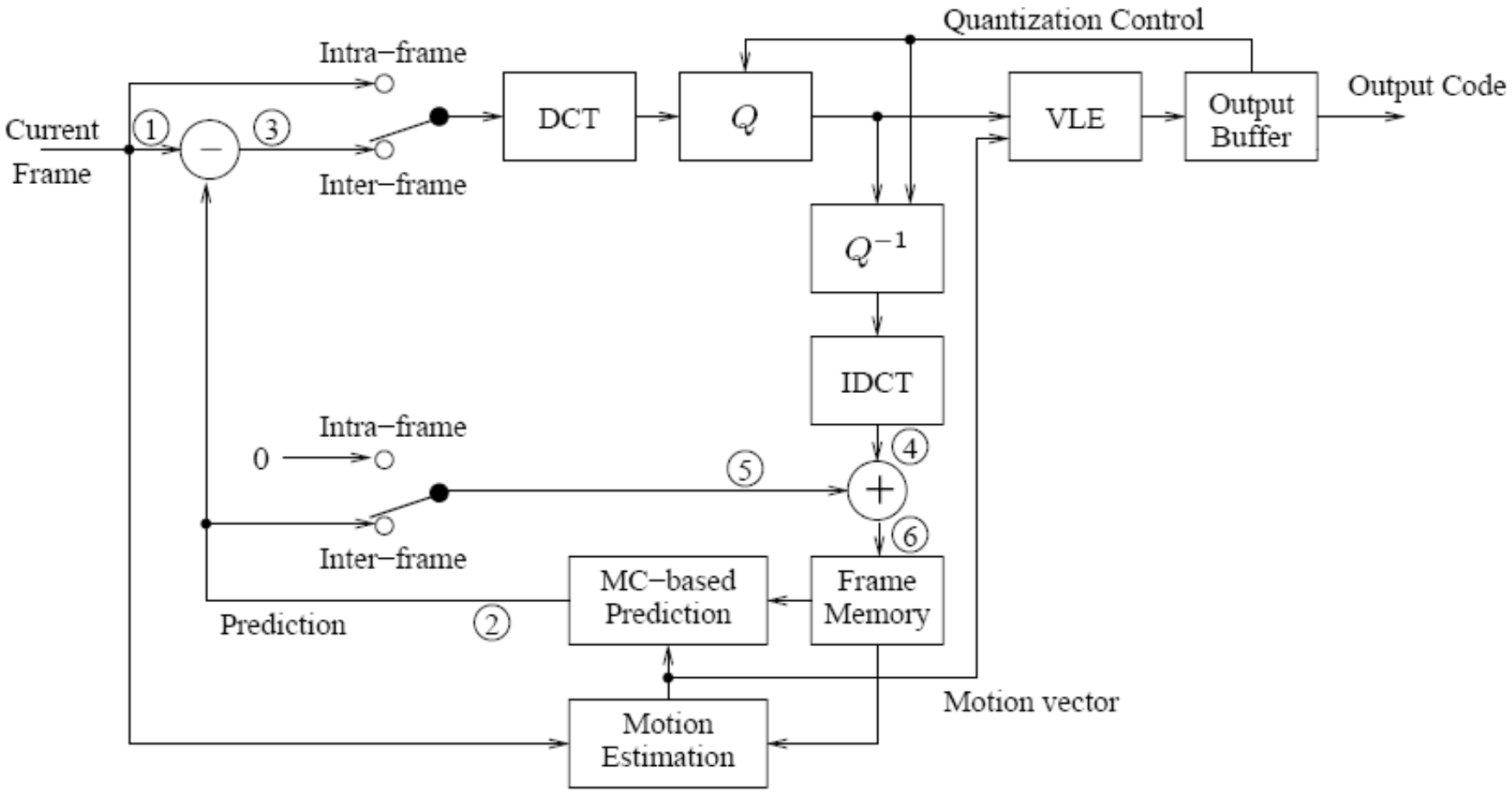
$$QDCT = round\left(\frac{DCT}{step\_size}\right) = round\left(\frac{DCT}{8}\right)$$

for all other coefficients:

$$QDCT = \left\lfloor \frac{DCT}{step\_size} \right\rfloor = \left\lfloor \frac{DCT}{2 * scale} \right\rfloor$$

scale : an integer in the range of [1, 31].

H.261 Encoder

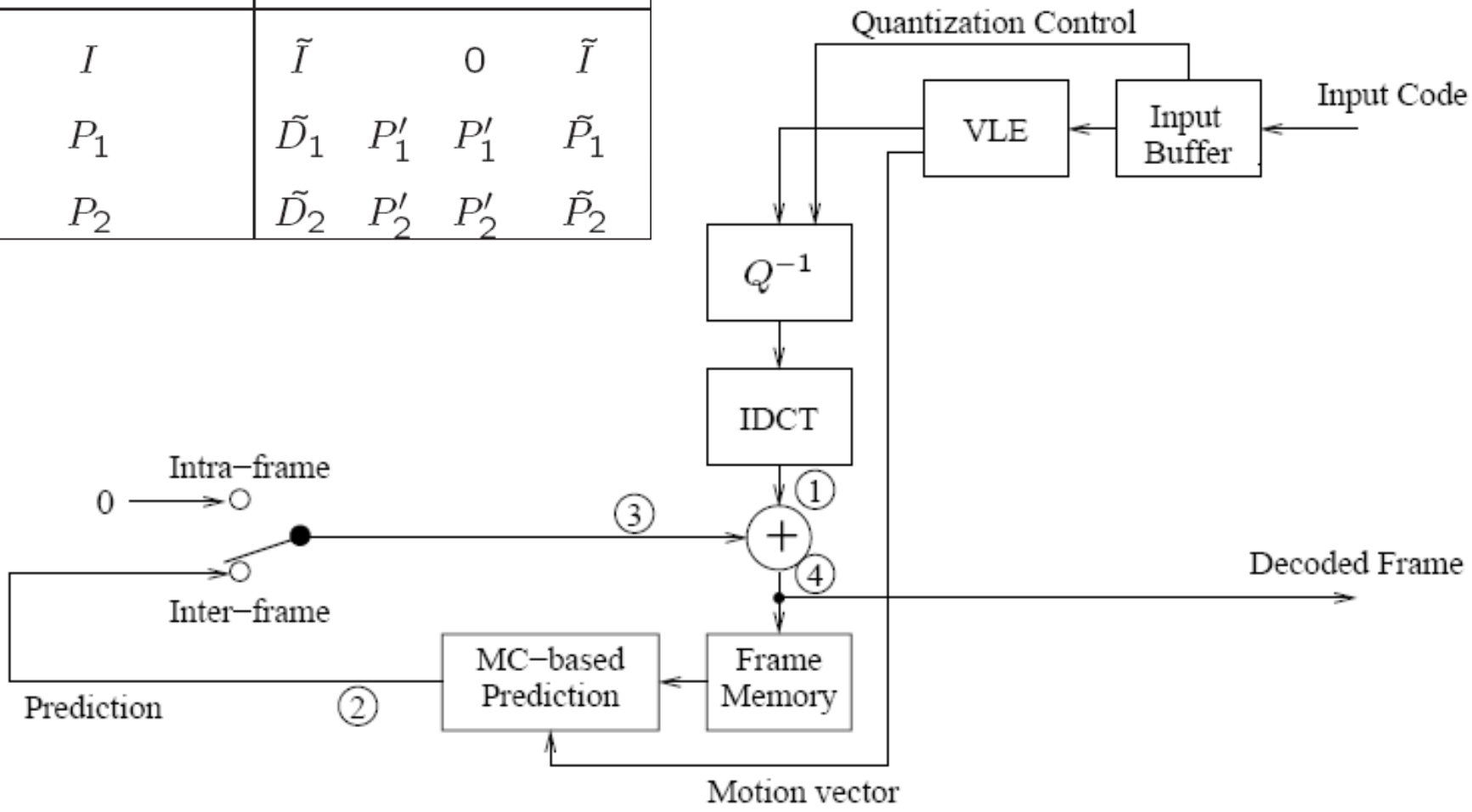


Note: decoded frames (not the original frames) are used as reference frames in motion estimation.

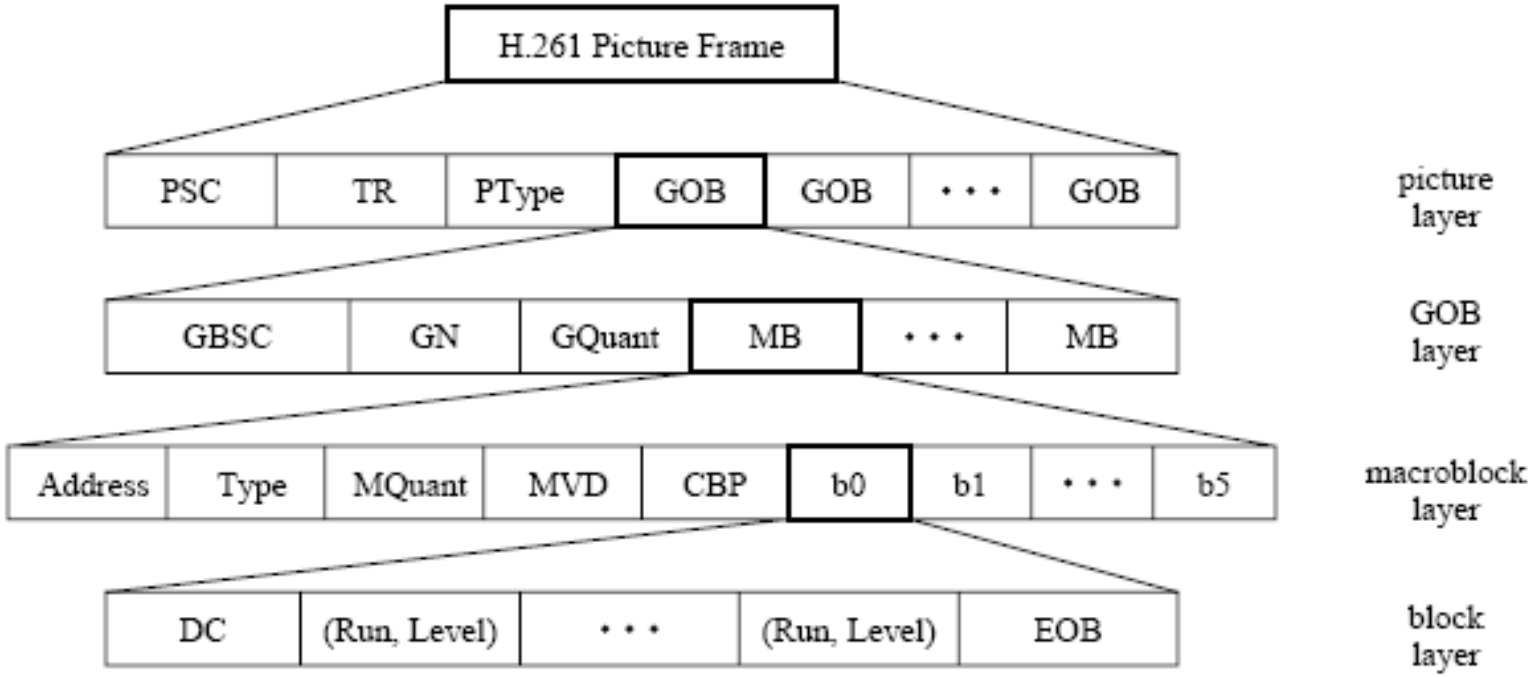
Current Frame	Observation Point					
	1	2	3	4	5	6
I	I			$\tilde{I}$	0	$\tilde{I}$
P_1	P_1	P_1'	D_1	$\tilde{D}_1$	P_1'	$\tilde{P}_1$
P_2	P_2	P_2'	D_2	$\tilde{D}_2$	P_2'	$\tilde{P}_2$

H.261 Decoder

Current Frame	Observation Point			
	1	2	3	4
I	$\tilde{I}$		0	$\tilde{I}$
P_1	$\tilde{D}_1$	P'_1	P'_1	$\tilde{P}_1$
P_2	$\tilde{D}_2$	P'_2	P'_2	$\tilde{P}_2$



Syntax of H.261 Video Bitstream.



PSC:	Picture Start Code	TR:	Temporal Reference
PType:	Picture Type	GOB:	Group of Blocks
GBSC:	GOB Start Code	GN:	Group Number
GQuant:	GOB Quantizer	MB:	Macro Block
MQuant:	MB Quantizer	MVD:	Motion Vector Data
CBP:	Coded Block Pattern	EOB:	End of Block

A Glance at Syntax of H.261 Video Bitstream

A hierarchy of four layers: Picture, Group of Blocks (GOB), Macroblock, and Block.

- The Picture layer: PSC (Picture Start Code) delineates boundaries between pictures. TR (Temporal Reference) provides a time-stamp for the picture.
- The GOB layer: H.261 pictures are divided into regions of 11x3 macroblocks, each of which is called a Group of Blocks (GOB).

A Glance at Syntax of H.261 Video Bitstream

- The Macroblock layer: Each Macroblock (MB) has its own Address indicating its position within the GOB, Quantizer (MQuant), and six 8x8 image blocks (4 Y, 1 Cb, 1 Cr).

- The Block layer: For each 8x8 block, the bit-stream starts with DC value, followed by pairs of length of zero-run (Run) and the subsequent non-zero value (Level) for ACs, and finally the End of Block (EOB) code.

Q&A