Video coding standards

Video signals represent sequences of images or frames which can be transmitted with a rate from 15 to 60 frames per second (fps), that provides the illusion of motion in the displayed signal.

Unlike images they contain the so-called temporal redundancy.

Temporal redundancy arises from repeated objects in consecutive frames of the video sequence. Such objects can remain, they can move horizontally, vertically, or any combination of directions (translation movement), they can fade in and out, and they can disappear from the image as they move out of view.

Temporal redundancy

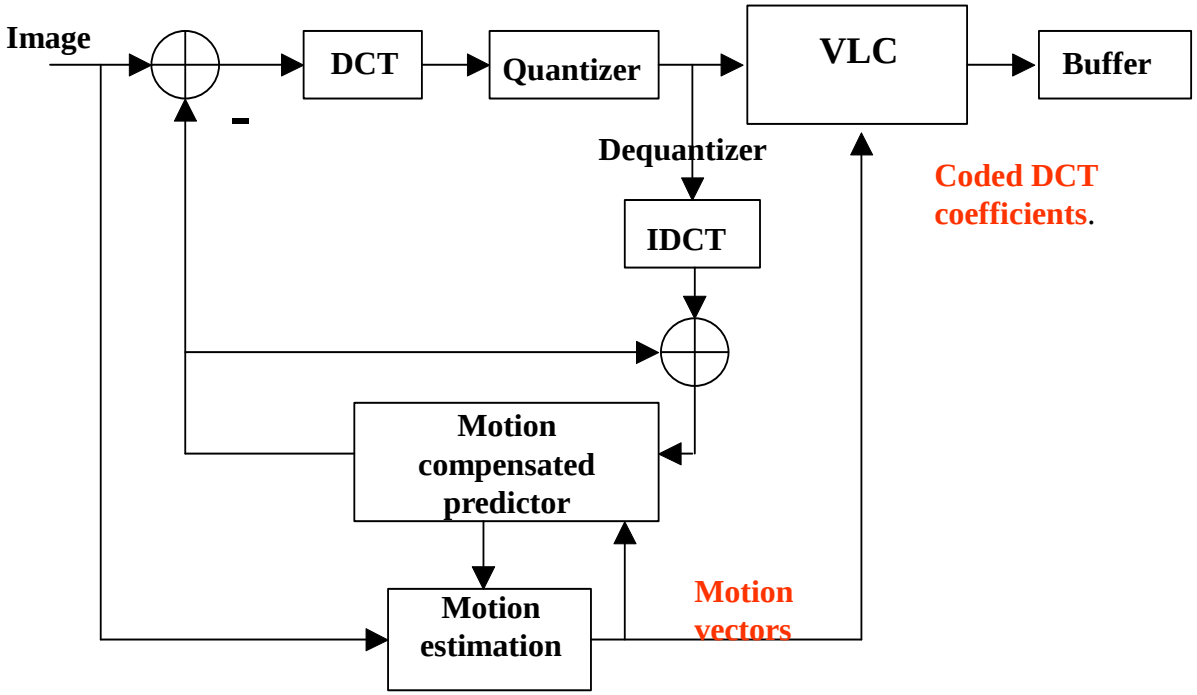
Motion compensation

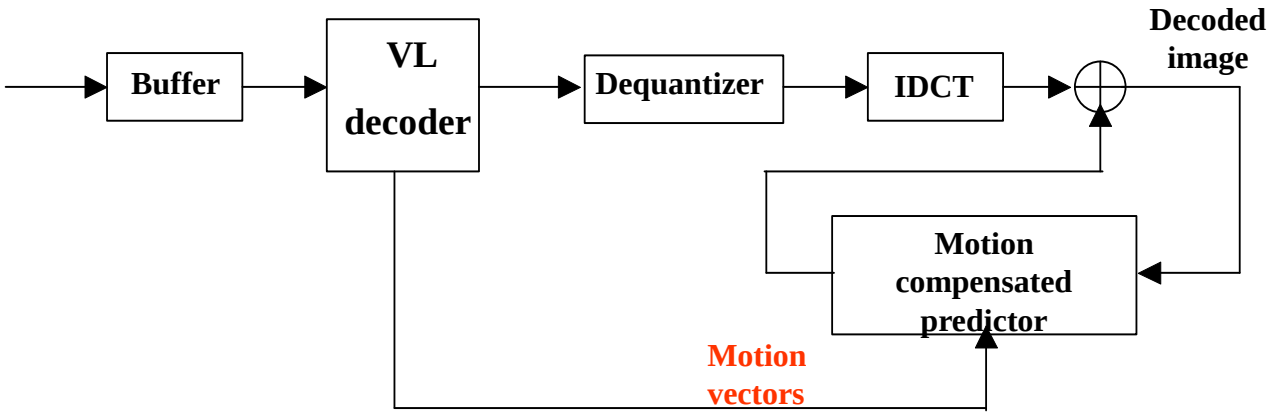
A motion compensation technique is used to compensate the temporal redundancy of video sequence.

The main idea of the this method is to predict the displacement of group of pixels (usually block of pixels) from their position in the previous frame.

Information about this displacement is represented by motion vectors which are transmitted together with the DCT coded difference between the predicted and the original images.

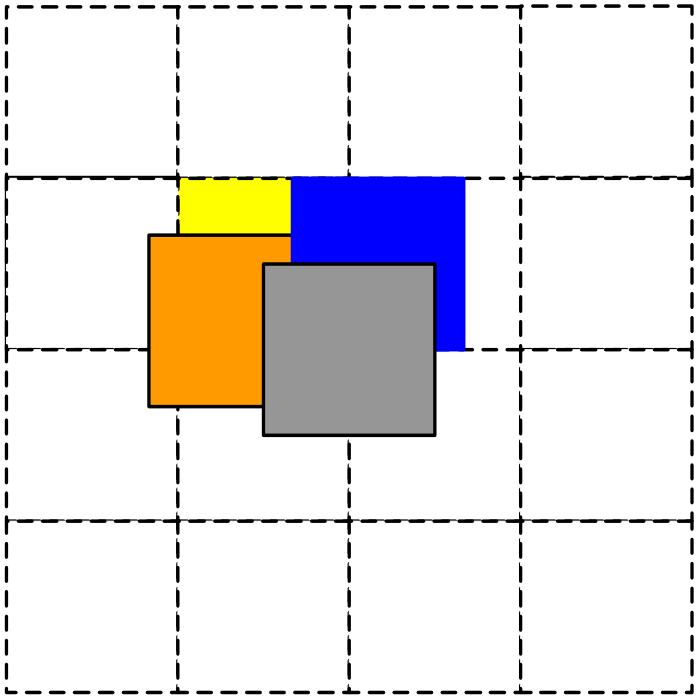
Motion compensation



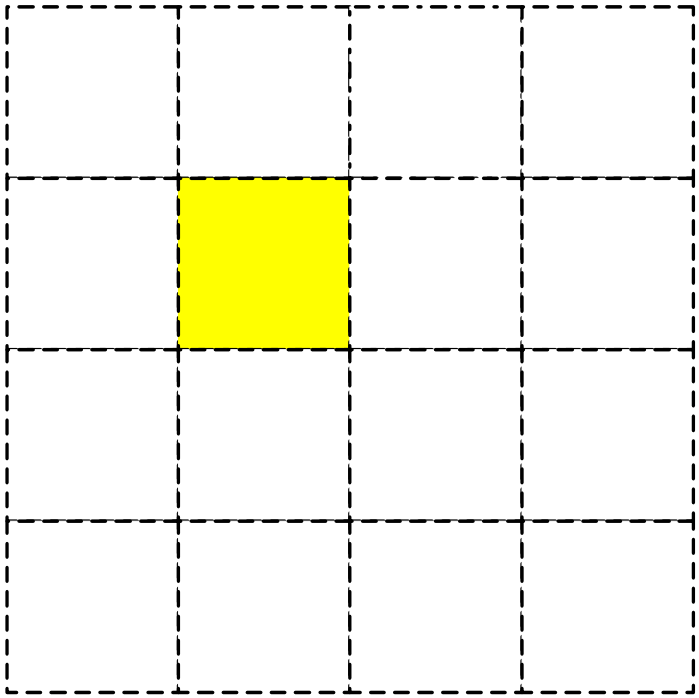


Motion compensation

Previous frame



Current frame

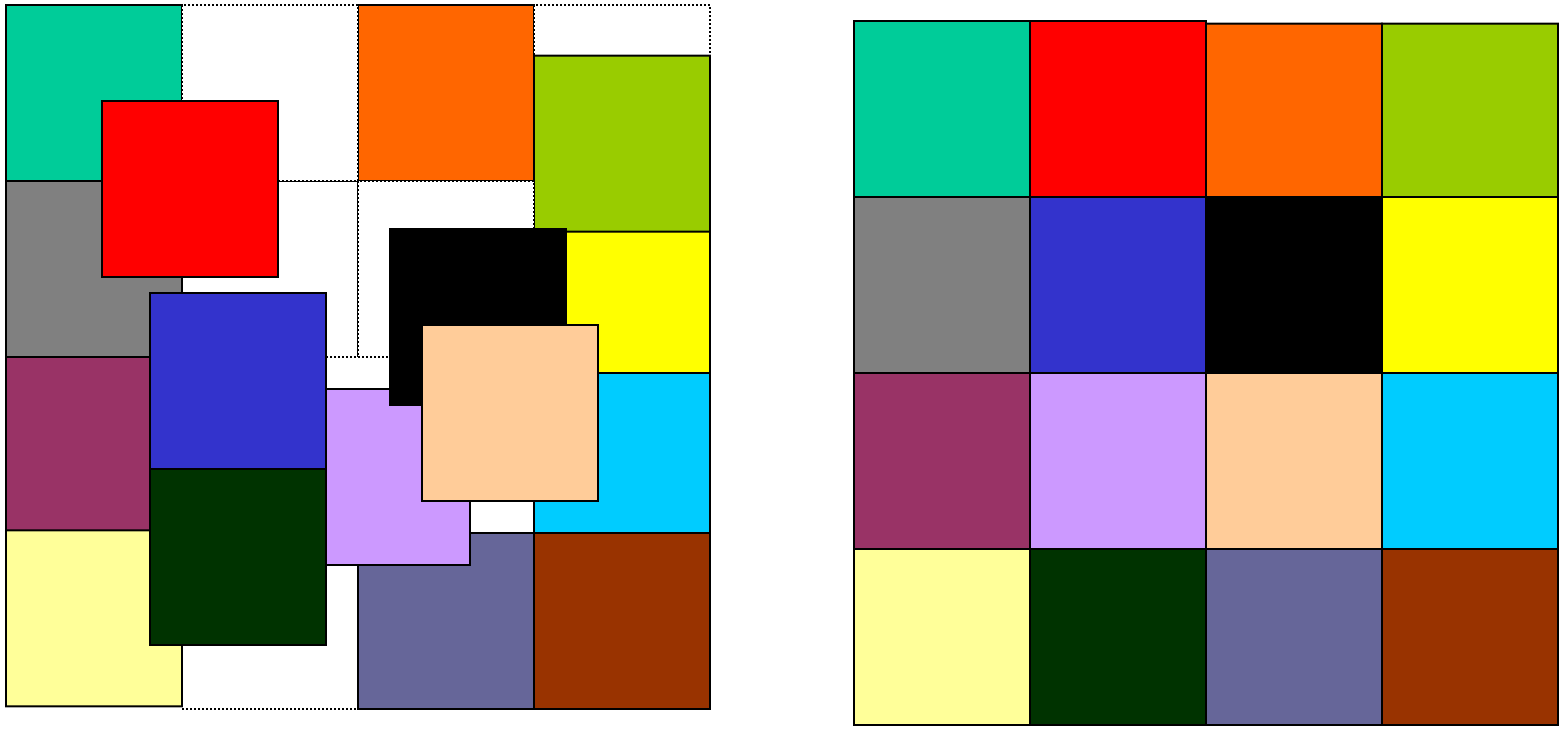


Set of macroblocks of the previous frame used to predict the selected macroblock of the current frame

Motion compensation

Previous frame

Current frame



Macroblocks of previous frame
used to predict current frame

Motion compensation

Each 16x16 pixel macroblock in the current frame is compared with a set of macroblocks in the previous frame to determine the one that best predicts the current macroblock.

The set of macroblocks in the previous frame is constructed as a set of shifts of the co-sited macroblock in the previous frame.

This set of macroblocks includes those within a limited region of the co-sited macroblock. To find the best matching macroblock we search for

$$\min_{\alpha,\beta} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left(x_c(m,n) - x_p(m+\alpha, n+\beta) \right)^2, \qquad (13.1)$$

where $x_c(m,n)$, $x_p(m,n)$ denote pixels of the current macroblock and the co-sited macroblock of the previous frame.

Motion compensation

Macroblock size is equal to $M \times N$ (usually 16x16) and usually we search for the best prediction for shifts

$$-M/2 \le \alpha \le M/2 \quad \text{and} \quad -N/2 \le \beta \le N/2.$$

When the best matching macroblock is found we construct a motion vector. For the i th macroblock of the current frame we transmit a pair of coordinates (α_i, β_i) that shows how we should translate the corresponding macroblock of the previous frame.

The motion vector $\mathbf{A} = (\alpha_1, \alpha_2, ..., \alpha_{N_B})$ contains coordinates α for all macroblocks and the motion vector $\mathbf{B} = (\beta_1, \beta_2, ..., \beta_{N_B})$ contains coordinates β for all macroblocks.

Motion compensation

Motion vectors are entropy coded and transmitted or stored as a part of the compressed data.

The difference between the current and the motion compensated frame is transformed using DCT, quantized, entropy coded and transmitted or stored together with coded motion vectors.

The main problem related with the motion compensation method is its high computational complexity. To minimize (13.1) we have to perform an exhaustive search over all admissible pairs α, β . If $-M/2 \le \alpha \le M/2$ and $-N/2 \le \beta \le N/2$ we search over $(M+1)(N+1)$ shifts.

If $M = N = 16$ we search over $17^2 = 289$ shifts.

Motion compensation

To speed up search procedure the logarithmic search is used.

Let $-4 \le \alpha \le 4$ and $-4 \le \beta \le 4$ or we have to consider

$9^2 = 81$ shifts.

Instead of this we first search over pairs: (0,4), (4,0), (0,0), (4,4), (-4,0), (0,-4), (-4,-4), (4,-4), (-4,4).

We obtain: $\mathbf{A}_1 = (\alpha_{11}, \alpha_{12}, ..., \alpha_{1N_B}),\ \mathbf{B}_1 = (\beta_{11}, \beta_{12}, ..., \beta_{1N_B})$

Then we use components of the found vectors as centers for the next search over pairs:

(0,2), (2,0), (0,0), (2,2), (-2,0), (0,-2), (-2,-2), (2,-2), (-2,2).

We obtain: $\mathbf{A}_2 = (\alpha_{21}, \alpha_{22}, ..., \alpha_{2N_B}),\ \mathbf{B}_2 = (\beta_{21}, \beta_{22}, ..., \beta_{2N_B})$

We use components of $\mathbf{A}_2, \mathbf{B}_2$ as centers for search over:

(0,1), (1,0), (0,0), (1,1), (-1,0), (0,-1), (-1,-1), (1,-1), (-1,1).

Motion compensation

At this step we obtain the motion compensation vectors **A**, **B**

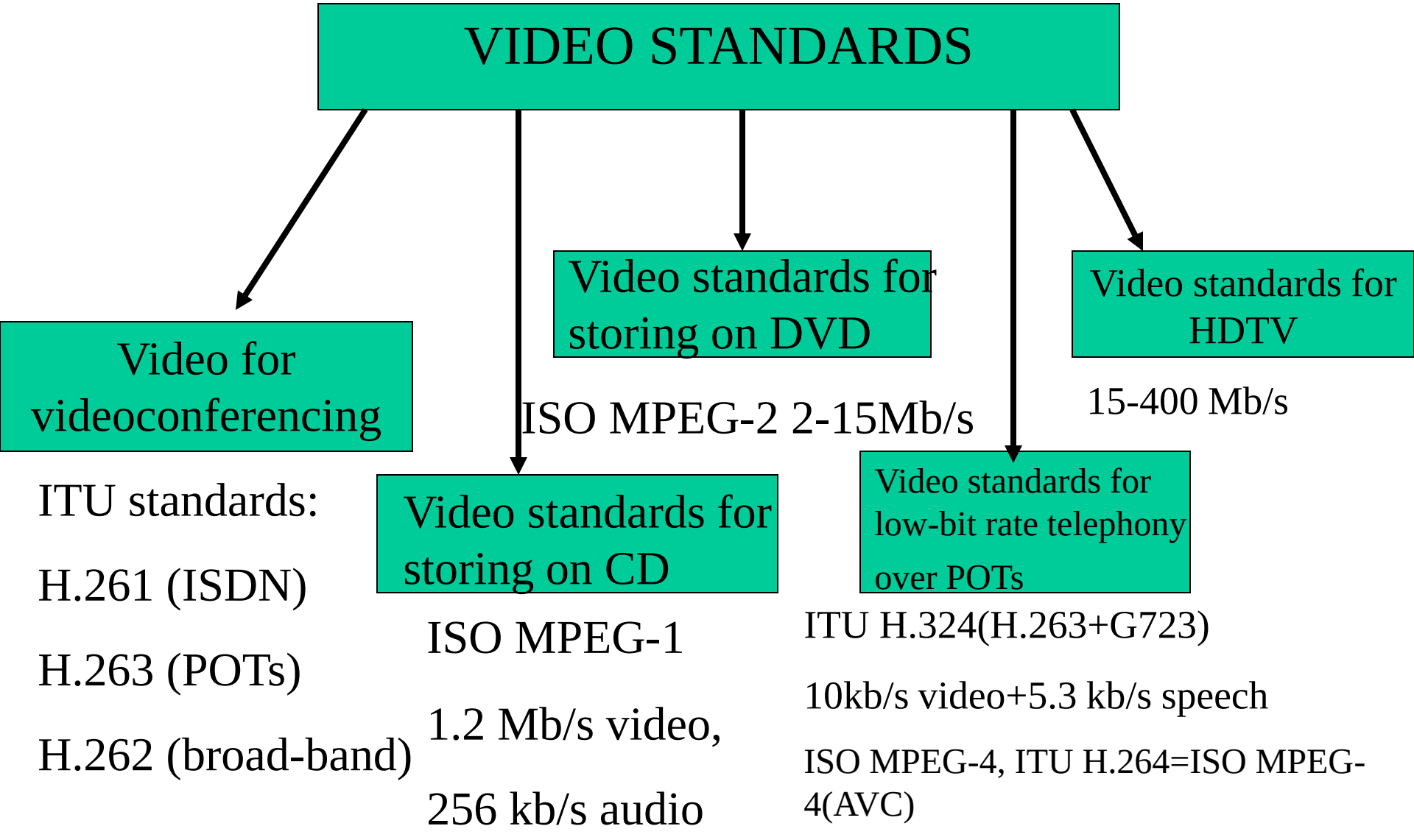
The described procedure searches over 27 shifts instead of 81.

The payment for the reducing of computational complexity of the search procedure is loss of its optimality.

However, this non-optimal search procedure usually does not lead to the significant loss in compression ratio for the given quality.

Object motion compensation

Overview of video standards



Common features of video standards

•All coders first determine the type of the frame using some criterion

INTRA or I-frame is coded and transmitted as an independent frame (as still images). An initial frame is always an I-frame. Other I-frames correspond to the frames where scenes change.

To encode the I-frames coders use a DCT on blocks 8x8 pixels and the corresponding part of the coder is the same as used for the JPEG coder.

•Consecutive frames which are modeled as changing slowly due to small motion of objects in the scene, are coded efficiently in the INTER mode using the motion compensation technique.

H.261 and its derivatives

H.261 is intended for ISDN teleconferencing .

H.262 is essentially the high-bit rate MPEG-2 standard

H.263 low bit rate video codec is intended for POTs teleconferencing at modem rates of 14.4-56 kb/s, where this rate includes video coding, speech coding, control information, and other logical channel data.

H.261 supports both CIF and the QCIF formats. It is intended for applications with small controlled amount of motion in a scene.

H.263 has the following improvements compared to H.261:

•Half-pixel motion compensation

•Improved VLC (arithmetic coding is used instead of the Huffman coding)

H.261 and its derivatives

•Advanced motion prediction mode, including overlapped block motion compensation

•A mode that combines a bidirectionally predicted picture with a normal forward predicted picture

The bidirectional prediction means that there are two types of predicted or INTER frames: P-frames and B-frames.

P-frames are predicted from the most recently reconstructed I or P frame.

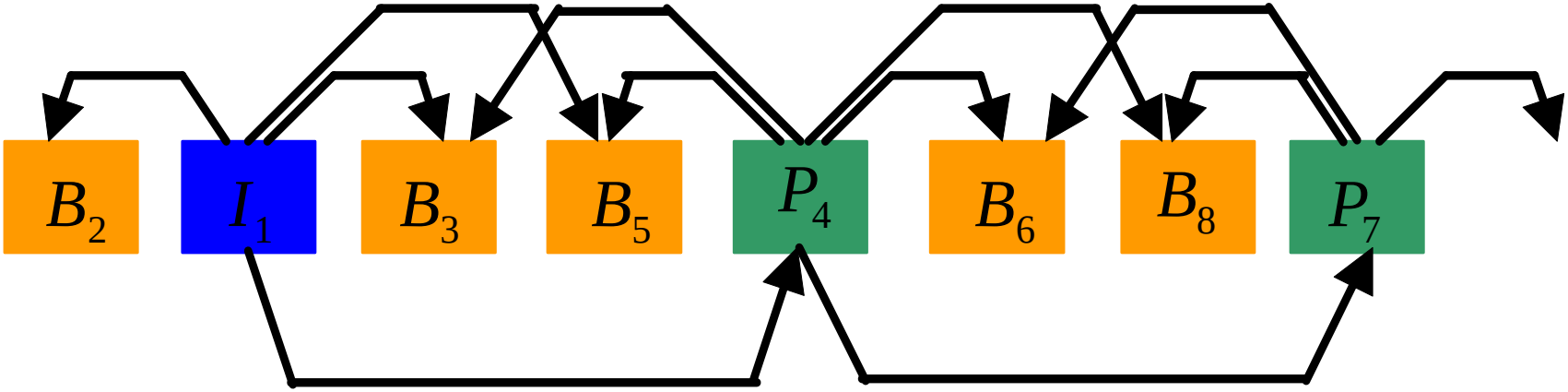
B-frames or bidirectional frames they are predicted from the closest two I or P frames, one in the past and one in the future.

•In addition H.263 supports a wider range of picture formats

(4CIF 704x576, 16CIF 1408x1152 and so on.)

Sequence of frames with bidirectional prediction

As stored or transmitted



As displayed

MPEG-1,MPEG-2, MPEG-4

The MPEG-1 standard is a true multimedia standard. It is optimized for storage of multimedia content on standard CD-ROM or applications at about 1.5 Mb/s.

It was designed to allow 74 minutes of digital video to be stored on CD. The supported picture formats are 352x288 at 25 fps and 352x240 pixels at 30 fps.

The video coding in MPEG-1 is very similar to the video coding of the H.26X series.

MPEG-2 is an extension of the MPEG-1 standard for digital compression of audio and video. It supports a wide range of bit rates. It efficiently codes interlaced video and provides tools for the scalable video coding.

Interlaced video coding

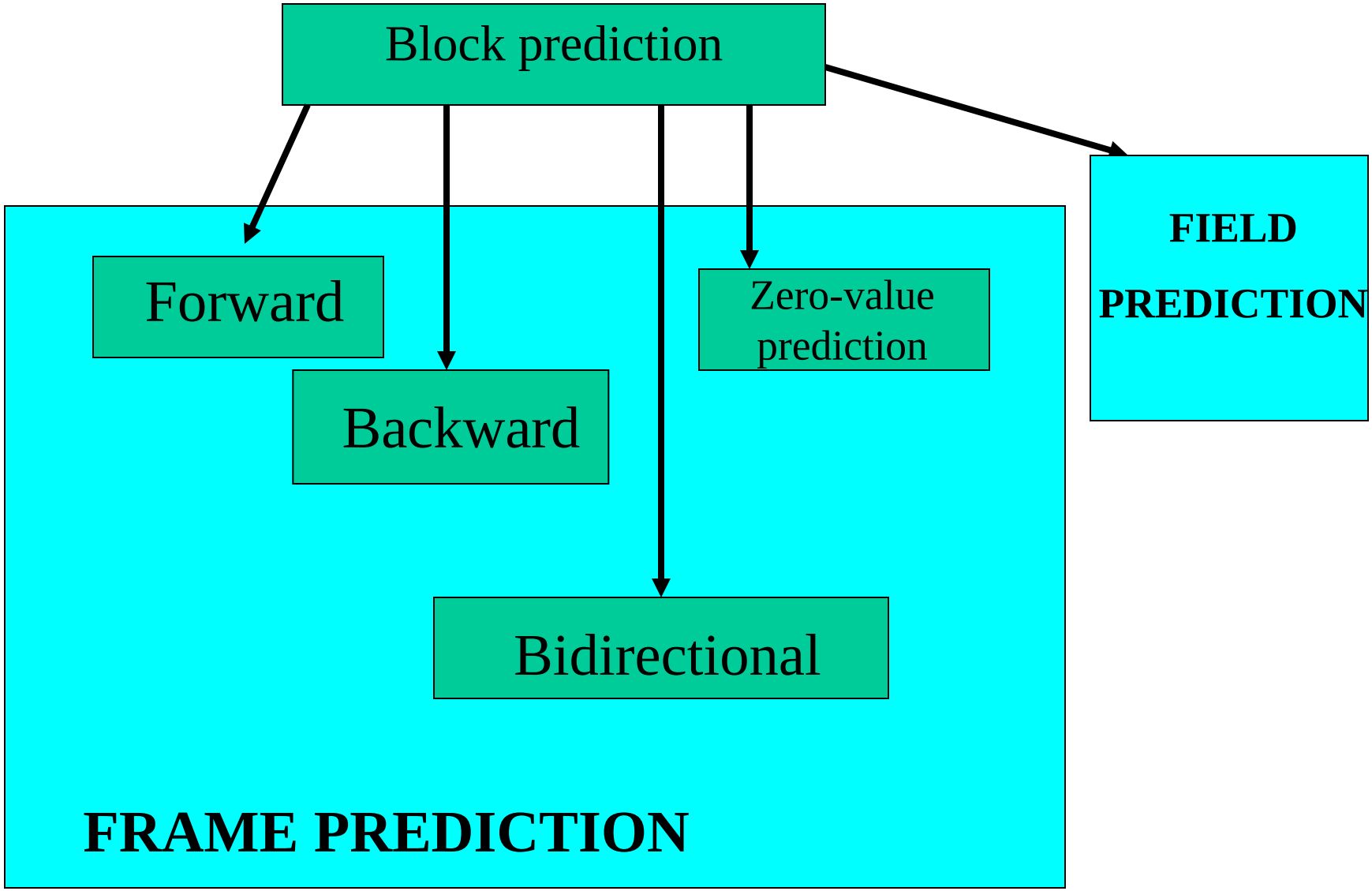
A movie is a sequence of frames displayed at a given rate. PAL TV is video displayed at 25 fps and NTSC is TV at 30 fps. Video at 25 or 30 fps is enough with human eye properties but on TV screen but image would be perceived to be flickering.

It was found that displaying the same frame in two parts (one field of odd lines and another field of even lines) and doubling the rate (60 ½ fps and 50 ½ fps) avoid flicker thanks to screen remanence.

TV is interlaced video. One frame contains fields from two instants in time.

In non-interlaced video all lines of the frame are sampled at the same instant in time. Non-interlaced video is also called progressively scanned or sequentially scanned video.

MPEG-2


Block prediction
Forward
Backward
Zero-value prediction
Bidirectional
FRAME PREDICTION
FIELD PREDICTION

MPEG-2

A profile is a subset of algorithmic tools.

A level identifies a set of constraints on parameters values (such as picture size, bit rate or number of layers supported by scalable profiles).

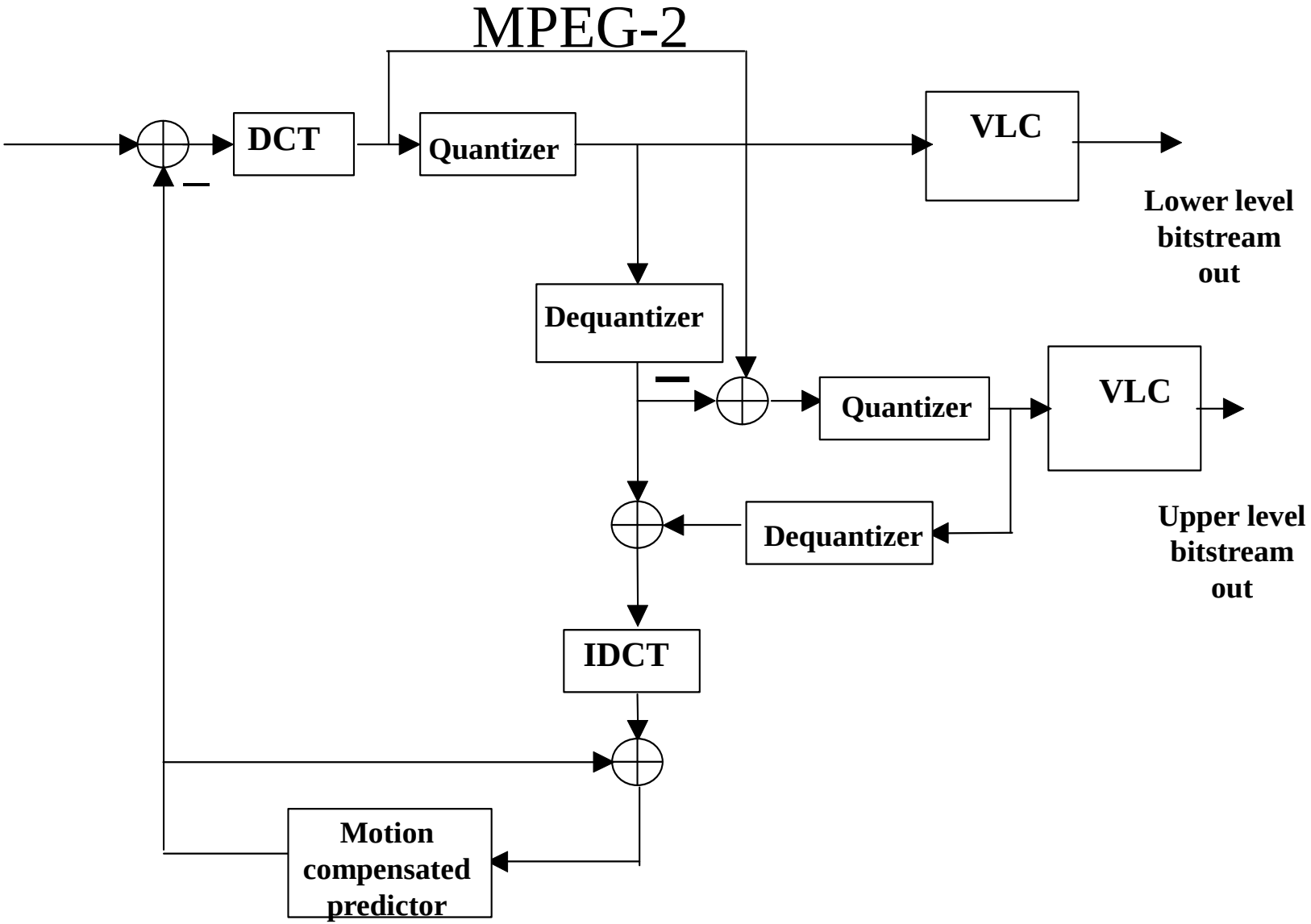
MPEG-2 supports two non-scalable profiles:

The simple profile uses no B-frames. It is suitable for low-delay applications such as videoconferencing.

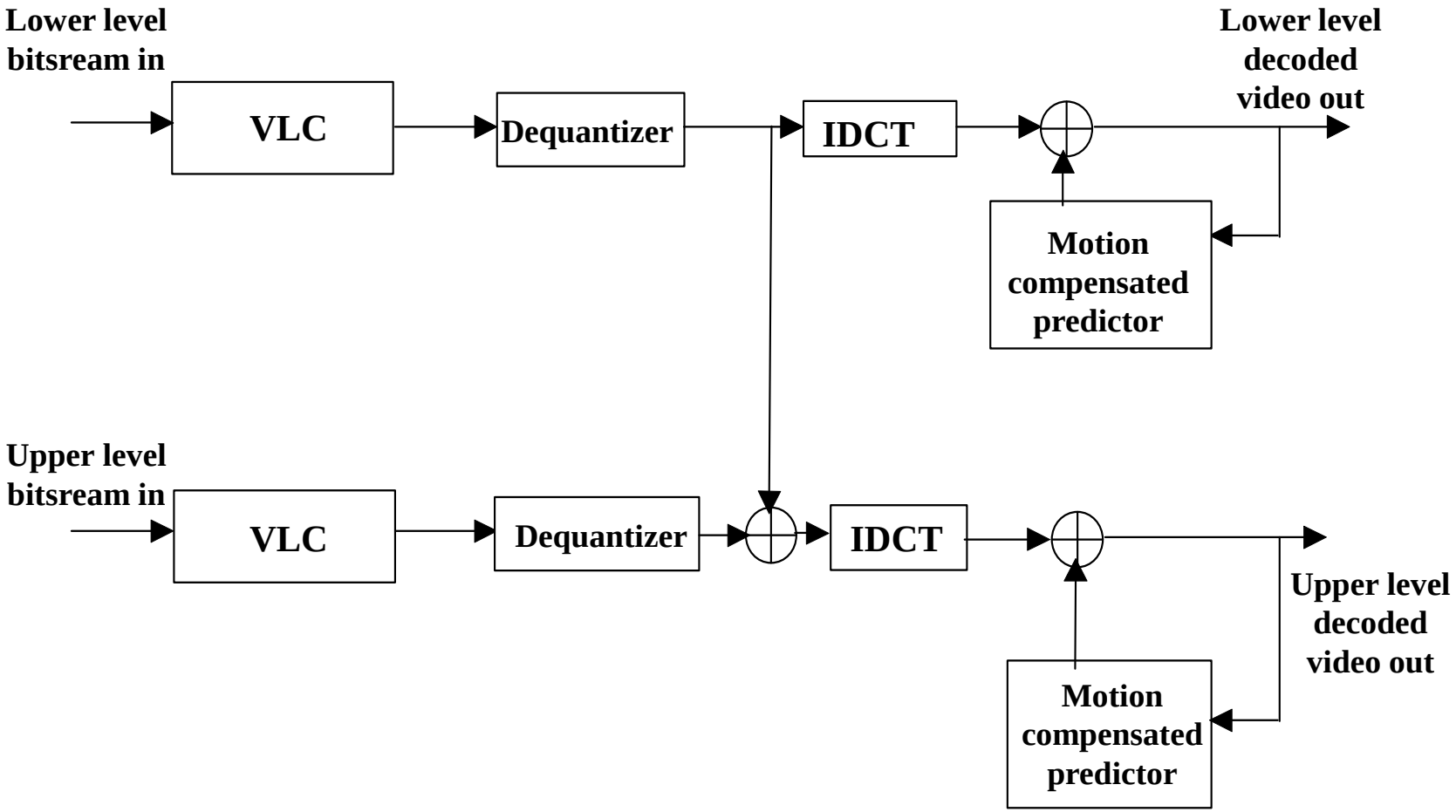
The main profile adds support for B-pictures and adds 120ms delay to allow picture reordering.

Scalable profiles: The SNR profile adds support for enhancement layers of DCT coefficient refinement. The total bitstream is structured in layers, starting with a base layer (it can be decoded itself) and adding refinement-layers to reduce quantization error.

MPEG-2


DCT
Quantizer
VLC
Lower level bitstream out
Dequantizer
Quantizer
VLC
Upper level bitstream out
Dequantizer
IDCT
Motion compensated predictor

MPEG-2


Lower level bitsream in
VLC
Dequantizer
IDCT
Lower level decoded video out
Motion compensated predictor
Upper level bitsream in
VLC
Dequantizer
IDCT
Upper level decoded video out
Motion compensated predictor

MPEG-4

It is optimized for three bitrate ranges:

1. Below 64 kb/s
2. 64-384 kb/s
3. 384 kb/s-4Mb/s

MPEG-4 provides support for both interlaced and progressively scanned video.

An MPEG-4 visual scene may consist of one or more objects.

Object (called visual object plane or VOP) is characterized by shape, motion and texture. It can be natural or synthetic and in the simplest case it can be a rectangular frame.

MPEG-4

The binary matrix representing the shape of a VOP is binary mask. In this mask pixels belonging to VOP are set to 1 and other pixels are set to 0. Binary mask is then split into binary alpha blocks (BAB) of size 16x16. The gray-scale mask is a matrix with either 8-bit integers (VOP) or zeros. It is also split into 16x16 alpha blocks.

If all pixels of BAB are zeros (a transparent block) or all belong to VOP (an opaque block) the block is flagged and coded in a special way. Binary shape for boundary BABs is encoded by using context arithmetic coding. The gray-scale information is coded by using motion compensation and DCT.

MPEG-4 provides 3 modes for encoding VOP:

1. INTRA or I-VOP; 2. Predicted VOP or P-VOP; 3. Bidirectional interpolated VOP or B-VOP

MPEG-4

Motion compensation is performed only for P- or B-VOPs.

For internal macroblocks 16x16 or 8x8 (in advanced mode) MC is done in the usual way.

For macroblocks that only partially belong to the VOP motion vectors are estimated using the modified block (polygon) matching technique.

The discrepancy of matching is given by the sum of absolute differences (SAD) computed for pixels belonging to the VOP. If the reference block is on the VOP boundary a repetitive padding technique assigns values to pixels outside the VOP. The SAD is computed using these padded pixels as well.

MPEG-4

MPEG-4 supports overlapped motion compensation. The motion vector for each pixel is constructed as the weighted sum of the current block motion vector and motion vectors for its 4 neighboring blocks.

For encoding texture information the standard 8x8 block-based DCT is used. To encode an arbitrary shaped VOP an 8x8 grid is superimposed on the VOP. Internal blocks are coded without modifications. Boundary blocks are extended into rectangular blocks using repetitive padding technique.

When the texture is the residual error after motion compensation the blocks are padded with zero-values.

MPEG-4

MPEG-4 provides a separate mode for encoding static texture information. It is based on wavelet coding, zero-tree algorithm and arithmetic coding.

A sprite coding is a very efficient method for compression of background video object. However, how automatically generate sprite image from raw video sequence is still an open issue.

Sprite-based coding is suitable for synthetic objects.

A sprite (mosaic) is an image composed from pixels belonging to a VOP visible throughout a video sequence. Background sprite is a still image consisting of all pixels belonging to the background. It can be transmitted only once at the beginning of transmission. At any time moment the background VOP can be extracted by warping/cropping this sprite.

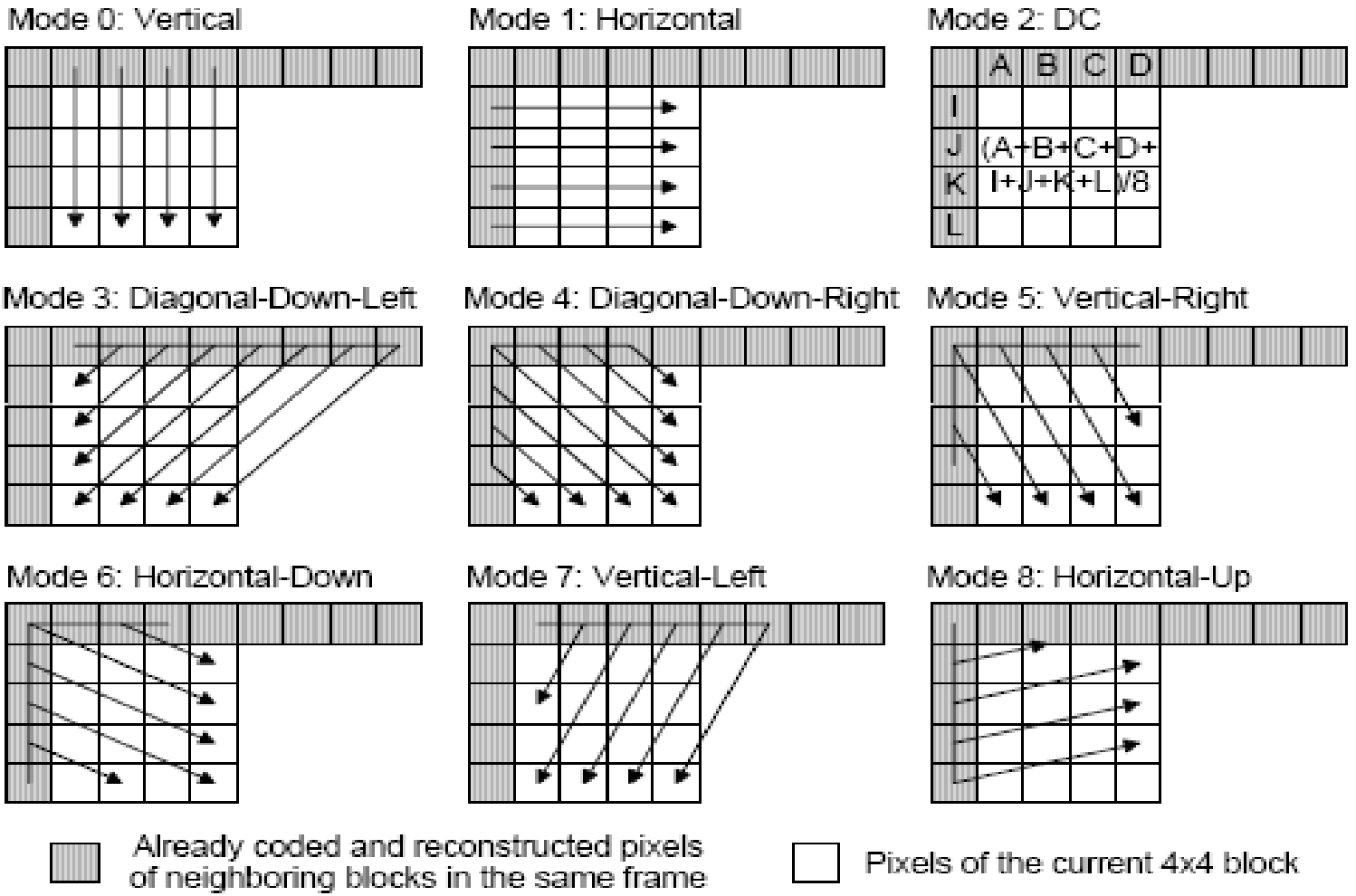
Sprite coding

H.264

- Combining transform coding with intra prediction in spatial domain (9 modes for 4x4 blocks and 4 modes for 16x16 blocks)
-

P(-1,-1)	P(-1,0)	P(-1,1)	P(-1,2)	P(-1,3)	P(-1,4)	P(-1,5)	P(-1,6)
P(0,-1)	B(0,0)	B(0,1)	B(0,2)	B(0,3)			
P(1,-1)	B(1,0)	B(1,1)	B(1,2)	B(1,3)			
P(2,-1)	B(2,0)	B(2,1)	B(2,2)	B(2,3)			
P(3,-1)	B(3,0)	B(3,1)	B(3,2)	B(3,3)			

H.264


Mode 0: Vertical
Mode 1: Horizontal
Mode 2: DC
A B C D
I
J
K
L
(A+B+C+D+
I+J+K+L)/8
Mode 3: Diagonal-Down-Left
Mode 4: Diagonal-Down-Right
Mode 5: Vertical-Right
Mode 6: Horizontal-Down
Mode 7: Vertical-Left
Mode 8: Horizontal-Up
Already coded and reconstructed pixels of neighboring blocks in the same frame
Pixels of the current 4x4 block

H.264

- Inter-frame prediction is based on hierarchical splitting of 16x16 macroblocks into blocks of smaller sizes. 16x16 – 16x8 – 8x16 –8x8 –8x4 –4x8 –4x4 . Smaller blocks are used for objects , larger blocks are used for background.
- $1/4^{th}$ pixel and $1/8^{th}$ pixel accuracy MC
- Multiple reference frames (for P-type a list of past frames is used, B-type=bi-predictive not bidirectional. Different reference frames for different partitions can be used.)
- Motion vectors are DPCM coded, (prediction of MV is constructed by using MVs of adjacent blocks)
- Skip-mode. Motion vectors=prediction vectors. Nonzero motion is accepted.
- DCT-II-like integer transform for 4x4 blocks

DCT-like integer transform

$$T=\begin{pmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & c \end{pmatrix} \quad a=0.5 \quad b=\sqrt{0.5}\cos(\pi/8) \quad c=\sqrt{0.5}\cos(3\pi/8)$$

$$Y = TXT^T = AXA^T \otimes B$$

$$A=\begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix}$$

$$B=\begin{pmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{pmatrix}$$

Inverse transform

$$X = C(Y \otimes B_I)C^T$$

$$C = \begin{pmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{pmatrix}$$

$$B_I = \begin{pmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{pmatrix}$$

DCT-like transform

- The multiplication by matrix A can be implemented in integer arithmetic by using only additions, subtractions, and shifts.
- Multiplication by matrices B and B_I is combined with the scalar quantization (dequantization)
- Lossless coding is implemented in two modes CABAC and CAVLC (based on the exponential Golomb coding)