

Video Coding

EE 446/646 Information and Coding Theory

Himali Singh	160003020
Khushboo Ahuja	160002024
Naman Singhal	160001038

What is a video?

- Visual multimedia source that combines a sequence of images to form a moving picture.
- The video transmits a signal to a screen and processes the order in which the screen captures should be shown.
- Videos usually have audio components that correspond with the pictures being shown on the screen.
- First developed for mechanical television systems, which were quickly replaced by cathode ray tube (CRT) systems which were later replaced by flat panel displays of several types.

Analog Video

vs

Digital Video

- Represents moving visual images using analog signals
- Originally, exclusively a live technology.
- Video tape recorder was later invented for recording analog video.
- In 1951, the first VTR captured live images from television cameras and saving the information onto magnetic video tape.
- However, video recorders were very costly. Prices gradually dropped over the years.

- Represents moving visual images using digital data
- It could not initially compete with analog video, due to early digital uncompressed video requiring impractically high bitrates.
- Practical digital video was made possible with video coding techniques.
- Digital video was later capable of higher quality and, eventually, much lower cost than earlier analog technology.

Characteristics of a Video File

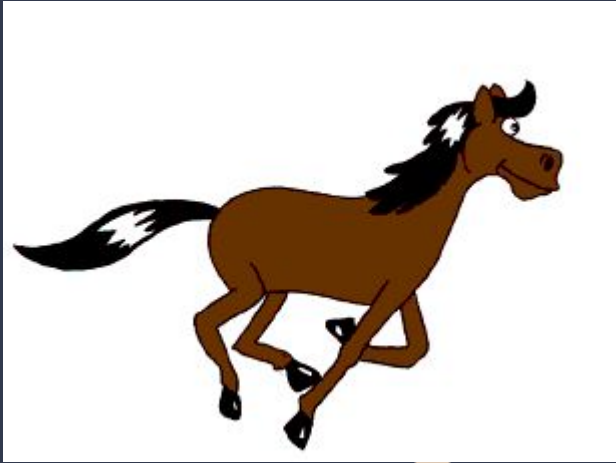
Display Resolution

- Number of distinct pixels in each dimension that can be displayed
- It is usually quoted as width × height, with the units in pixels.
- For eg., a popular resolution, 1080p or Full HD, has resolution 1920 x 1080 indicating the width of 1920 pixels and the height of 1080 pixels.

Aspect Ratio

- The ratio of the width of an image to its height
- It is commonly expressed as two numbers separated by a colon, as in 16:9.
- Widely used aspect ratios include 1.85:1 and 2.39:1 in film photography, 4:3 and 16:9 in television, and 3:2 in still camera photography.

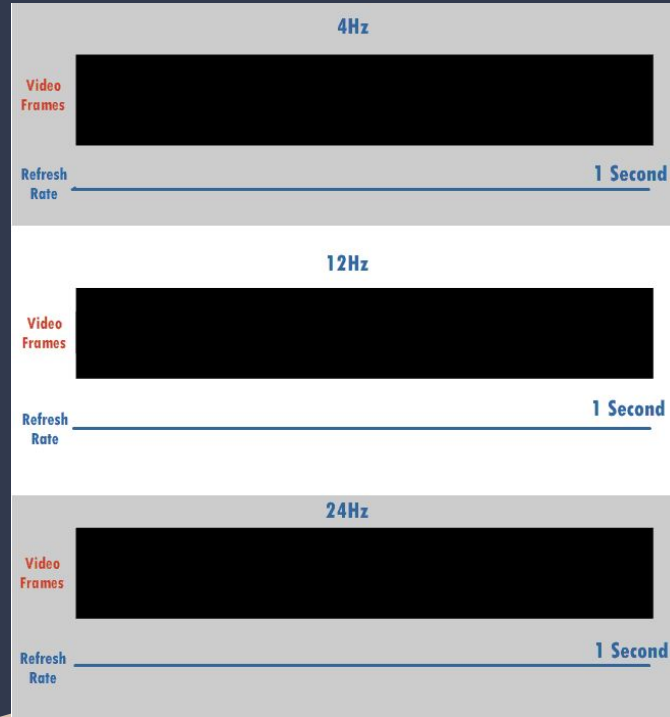
Frame & Frame Rate



This animated cartoon is displayed at 12 drawings per second, and the fast motion is on the edge of being objectionably jerky.

- In filmmaking, video production, animation, and related fields, a **frame** is one of the many still images which compose the complete moving picture.
- **Frame rate** is the frequency (rate) at which consecutive frames appear on a display.
- Frame rate is expressed in frames per second or FPS.
- The human visual system can process 10 to 12 images per second and perceive them individually, while higher rates are perceived as motion.
- The industry standard of frame rate for films is 24 FPS

Refresh Rate



Refresh Rate - 4Hz, 12Hz, 24Hz; Frame Rate - 24 FPS

- Number of times in a second that a display hardware updates its buffer.
- The refresh rate includes the repeated drawing of identical frames, while frame rate measures how often a video source can feed an entire frame of new data to a display.
- For eg., most movie projectors advance from one frame to the next one 24 times each second. But each frame is illuminated two or three times before the next frame is projected. As a result, the movie projector runs at 24 frames per second, but has a 48 or 72 Hz refresh rate.

Color Model

- A color model is an abstract mathematical model describing the way colors can be represented as tuples of numbers, typically as three or four values or color components.
- For eg. YCbCr color model is used for digital video.
- Other examples include RGB, YIQ, YUV color models.

Color Depth

- The number of bits used to indicate the color of a single pixel, in a bitmapped image or video framebuffer, or the number of bits used for each color component of a single pixel.
- When referring to a pixel, the concept is bits per pixel (bpp) defined as **bits per pixel (bpp)** and when referring to a color component, the concept is defined as **bits per component (bpc)**.



24 bit, 16,777,216 colors, 98 KB



8 bit, 256 colors, 37 KB



4 bit, 16 colors, 13 KB



2 bit, 4 colors, 6 KB



1 bit, 2 colors, 4 KB

Same image on five different color depths (bits)

Image Source : Wikipedia

Bit Rate & Bits Per Pixel (BPP)

- **Bit rate** is the number of bits that are conveyed or processed per unit of time, thus is a measure of the rate of information content of the digital video stream.
- Bit rate corresponds directly to the quality of the video as bit rate is proportional to every property that affects the video quality.
- The video size is proportional to the bit rate and the duration.
- Video compression is used to greatly reduce the bit rate while having a lesser effect on quality.
- **Bits per pixel (BPP)** is a measure of the efficiency of compression.
- A true-color video with no compression at all may have a BPP of 24 bits/pixel. Applying jpeg compression on every frame can reduce the BPP to 8 or even 1 bits/pixel. Applying video compression algorithms like MPEG1, MPEG2 or MPEG4 allows for fractional BPP values.

Motivation – Why do we need Video Coding?

Uncompressed digital video delivers maximum quality, but with a very high data rate. Because of the large amount of storage and bandwidth needed to record and convey raw video, a method was needed to reduce the amount of data used to represent the raw video. Since then, engineers and mathematicians have worked on developing methods to compress the video for practical usage.

In 1974, discrete cosine transform (DCT) compression was introduced by Nasir Ahmed, T. Natarajan and K. R. Rao. During the late 1980s, a number of companies began experimenting with DCT lossy compression for video coding, leading to the development of the H.261 standard. H.261 was the first practical video coding standard, and was developed by a number of companies, including Hitachi, PictureTel, NTT, BT, and Toshiba, among others. Since H.261, DCT compression has been adopted by all the major video coding standards that followed.

Major Video coding initiatives and standards

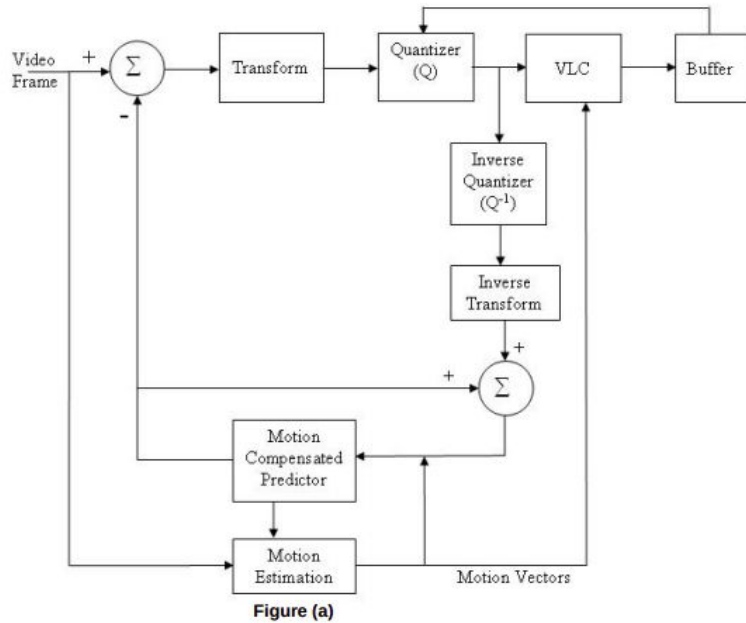
- Video coding for video teleconferencing, which has led to the ITU standards – H.261 for ISDN video conferencing, H.263 for very low bitrate and Plain Old Telephone Systems (POTS) video conferencing and the latest H.264 for video telephony and video streaming in wireless applications.
- Video coding for storing movies on CD-ROM, with a bit-rate up to 1.5 M bits/sec, of which 1.2 Mbits/sec is allocated to video and 256 Kbits/sec is allocated to the audio. This is addressed by ISO MPEG-1.
- Video coding for broadcast and storing video on digital video disks (DVD) with an order of 2-15 Mbits/sec allocated to audio and video coding, which led to the ISO-MPEG-2 standard. This, standard also addressed High Definition Television (HDTV) applications.
- Coding of separate audio-visual objects, both natural and synthetic, which led to the ISO-MPEG-4 standard.
- Coding of multimedia metadata, i.e., the data describing the features of multimedia data, which is being addressed in the upcoming MPEG-7 standard

Introduction to Video Coding

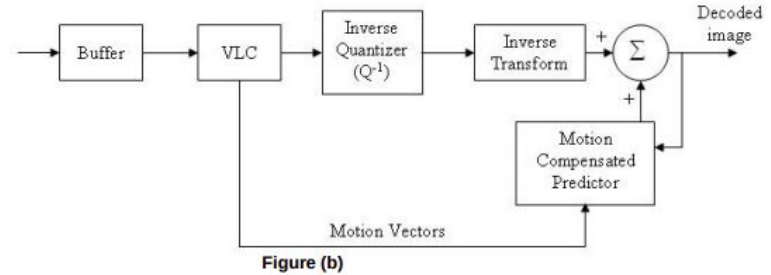
A wide range of emerging applications, such as conversational video like video telephone , video conferencing through wired and wireless medium; streaming video, such as video on demand; digital TV/HDTV broadcasting , image / video database services, CD/DVD storage etc, demand significant amount of data compression.

Video Coding is the process through which this compression can be achieved. In the subsequent presentation, we aim to introduce the process of video coding and decoding along with a working example using python code and the various standards of video coding.

Hybrid Video Codec



Video Encoder



Video Decoder

Hybrid Video Codec

- The predictive technique is used to temporally predict the current frame from the previous one(s) that must be stored.
- The temporal prediction is based on the assumption that the consecutive frames in a video sequences exhibit very close similarity.
- The predicted frame generated by the exploitation of temporal redundancy is subtracted from the incoming video frame, pixel by pixel and the difference is the error image.
- The error image goes through transforms as in still image codecs to exploit spatial redundancy.
- The transformed coefficients are quantized and entropy coded before adding to the bit stream.

Hybrid Video Codec

- The encoder has a built in decoder too so as to reconstruct the error frame, which will not be exact, because of the quantizer.
- The error frame is added to the predicted frame to generate the buffer.
- The motion estimation block determines the displacement between the current frame and the stored frame.
- The displacements so computed are applied on the stored frame in the motion compensation unit to generate the predicted frame.

Discrete Cosine Transform (DCT)

- Most popular transform for image compression due to its high energy compaction
- Used to map an image space into frequency domain
- Generally, $N = 8$ is used for image compression applications where image blocks of size 8×8 are converted into the frequency domain.
- **Inverse Discrete Cosine Transform (IDCT)** is used for inverse transform
- The equation for 2D $N \times N$ DCT and IDCT is given by-

$$F(u, v) = \frac{2}{N} C(u) C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \cos \left[\frac{\pi(2x+1)u}{2N} \right] \cos \left[\frac{\pi(2y+1)v}{2N} \right]$$

for $u = 0, \dots, N-1$ and $v = 0, \dots, N-1$

$$\text{where } N = 8 \text{ and } C(k) = \begin{cases} 1/\sqrt{2} & \text{for } k = 0 \\ 1 & \text{otherwise} \end{cases}$$

DCT

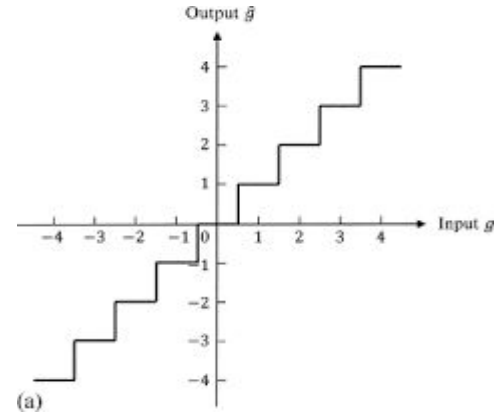
$$f(x, y) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u) C(v) F(u, v) \cos \left[\frac{\pi(2x+1)u}{2N} \right] \cos \left[\frac{\pi(2y+1)v}{2N} \right]$$

for $x = 0, \dots, N-1$ and $y = 0, \dots, N-1$ where $N = 8$

IDCT

Quantization

- Used to reduce a range of numbers to a single small value, so that we can use less bits to represent a large number.
- A quantizer step size (Q) and a round function is defined
- $x_q = \text{round}(x/Q)$
- Larger Q leads to more compression but at the cost of larger distortion.
- e.g. for x in $[0,255]$, we need 8 bits and have 256 color values – with $Q=64$ we only have 4 levels and only need 2 bits
- **Inverse Quantization** (dequantize) is used to reconstruct original value. But as we use $\text{Round}()$ function to get a nearest integer value, so reconstructed value will not be the same with original value.
- In general if we carefully design $Q[i,j]$, visual quality will not be affected.



An example of uniform quantizer function

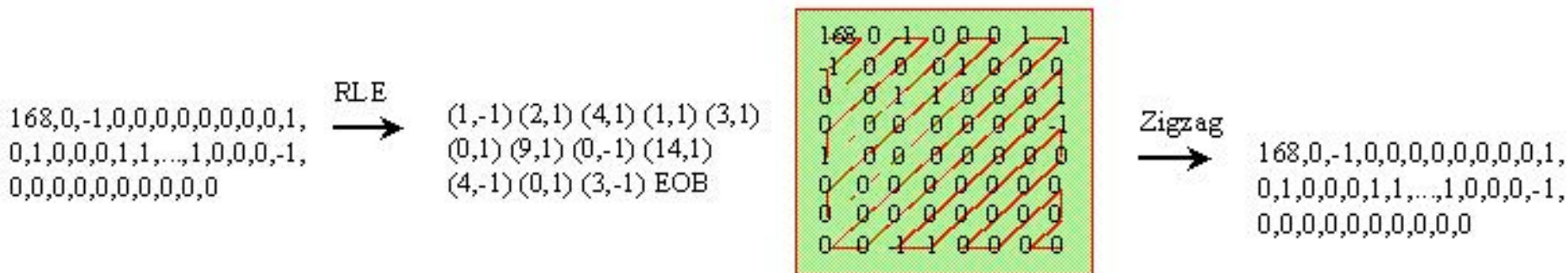
Example of application of 8 x 8 DCT and Quantization with quantizer factor 4 to an 8 x 8 block of an image.

88 84 83 84 85 86 83 82	DCT →	67 51 -6 2 -2 0 5 -5
86 82 82 83 82 83 83 81		-4 1 2 1 5 1 -3 0
82 82 84 87 87 87 81 84		2 3 4 6 -2 2 1 5
81 86 87 89 82 82 84 87		-3 -1 0 2 0 -2 2 -4
81 84 83 87 85 89 80 81		4 3 1 -1 -2 1 -3 1
81 85 85 86 81 89 81 85		1 -2 0 -3 2 -1 1 1
82 81 86 83 86 89 81 84		3 0 -1 0 -1 -1 0 -2
88 88 90 84 85 88 88 81		-1 -1 -5 5 2 -2 2 0

67 51 -6 2 -2 0 5 -5	Quant →	168 0 -1 0 0 0 1 -1
-4 1 2 1 5 1 -3 0		-1 0 0 0 1 0 0 0
2 3 4 6 -2 2 1 5		0 0 1 1 0 0 0 1
-3 -1 0 2 0 -2 2 -4		0 0 0 0 0 0 0 -1
4 3 1 -1 -2 1 -3 1		1 0 0 0 0 0 0 0
1 -2 0 -3 2 -1 1 1		0 0 0 0 0 0 0 0
3 0 -1 0 -1 -1 0 -2		0 0 0 0 0 0 0 0
-1 -1 -5 5 2 -2 2 0		0 0 -1 1 0 0 0 0

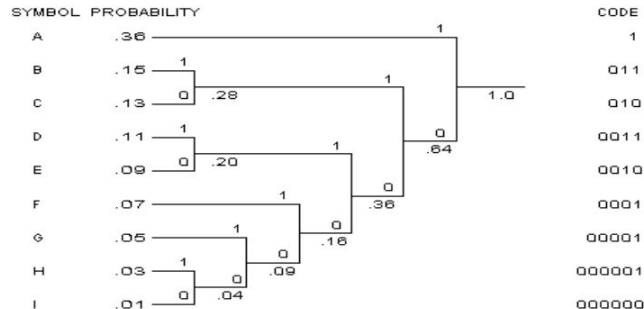
Zig Zag Scan and Run Length Encoding (RLE)

- Zig zag scan and RLE are often applied after the quantization step.
- After DCT and quantization most AC values will be zero. By using zig-zag scan we can gather even more consecutive zeros, then we use RLE to gain compression ratio.
- After zig-zag scan, many zeros are together now, so we encode the bitstream as (skip,value) pairs, where skip is the number of zeros and value is the next non-zero component.
- Zig-zag scan and RLE are only used in AC coefficient. DPCM is used for DC coefficient.



Entropy Encoding

- It is a lossless data compression scheme
- Entropy encoding can be categorized into fixed length coding (FLC) and variable length coding (VLC)
- VLC has added advantages over FLC as it uses shorter codes for more likely outcomes leading to more compression.
- In VLC, optimal code length for a symbol is $-\log_b P$, b=number of symbols, P=probability of the input symbol.
- Two of the most common entropy encoding techniques are Huffman coding and arithmetic coding.



Huffman Coding Example

Fixed length coding:

Huffman code:

Entropy

Redundancy of the Huffman code:

$$R_{\text{fixed}} = 4 \text{ bits/symbol}$$

$$R_{\text{Huffman}} = 2.77 \text{ bits/symbol}$$

$$H(X) = 2.69 \text{ bits/symbol}$$

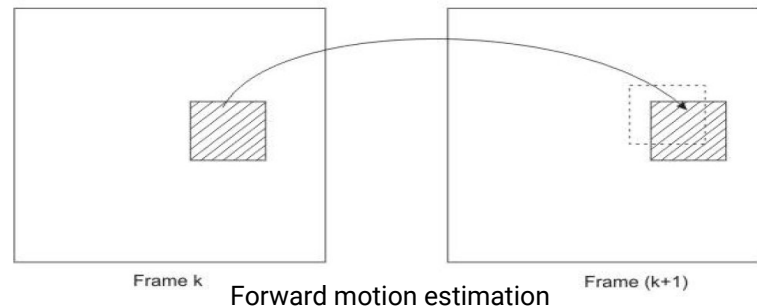
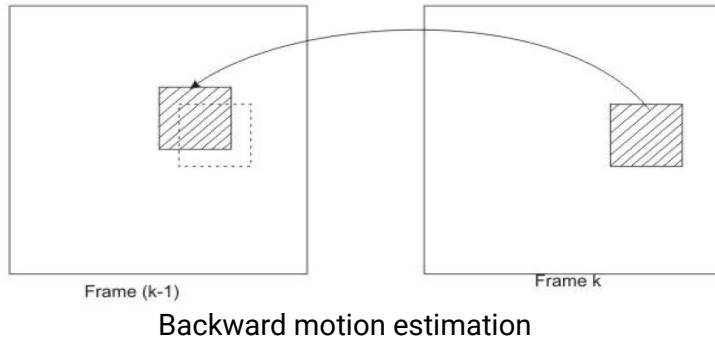
$$\rho = 0.08 \text{ bits/symbol}$$

Motion Estimation and Motion Compensation

- The **motion estimation** block computes the displacement between the current frame and a stored past frame that is used as a reference. Usually the immediate past frame is considered to be the reference.
- Considering a pixel from the current frame with its neighbourhood as the candidates, the best matching position in the reference frame is determined.
- The difference in position between the candidates and its match is defined as the displacement or motion vector. The **motion vector** has both horizontal and vertical components of displacement.
- The **motion compensation** block predicts the current frame by applying the displacements corresponding to the motion vectors on the reference frame.
- Here, it is assumed that the displacement is purely translational. Any scaling, rotation or 3-D deformation is neglected. The assumptions are reasonable for movement of objects over one frame in block based motion estimation.

Backward and Forward motion estimation

- In **backward motion estimation**, the current frame (Frame k) is considered as the candidate frame and a past frame (Frame $(k-1)$) is considered as the reference frame. The search for motion vectors is backward which leads to forward motion prediction.
- In **forward motion estimation**, the reference frame (Frame $(k+1)$) chosen is a frame that appears later than the candidate frame in temporal ordering. Here, the candidate frame is one of the past stored frames. The search for motion vectors is forward which leads to backward motion prediction.

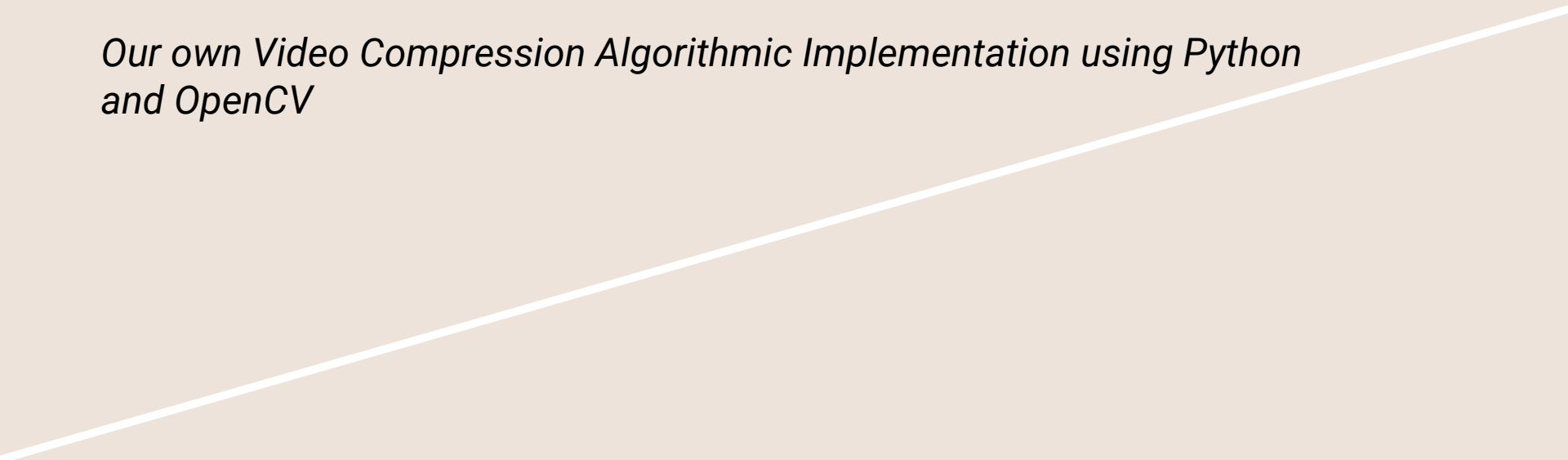


Motion estimation approaches

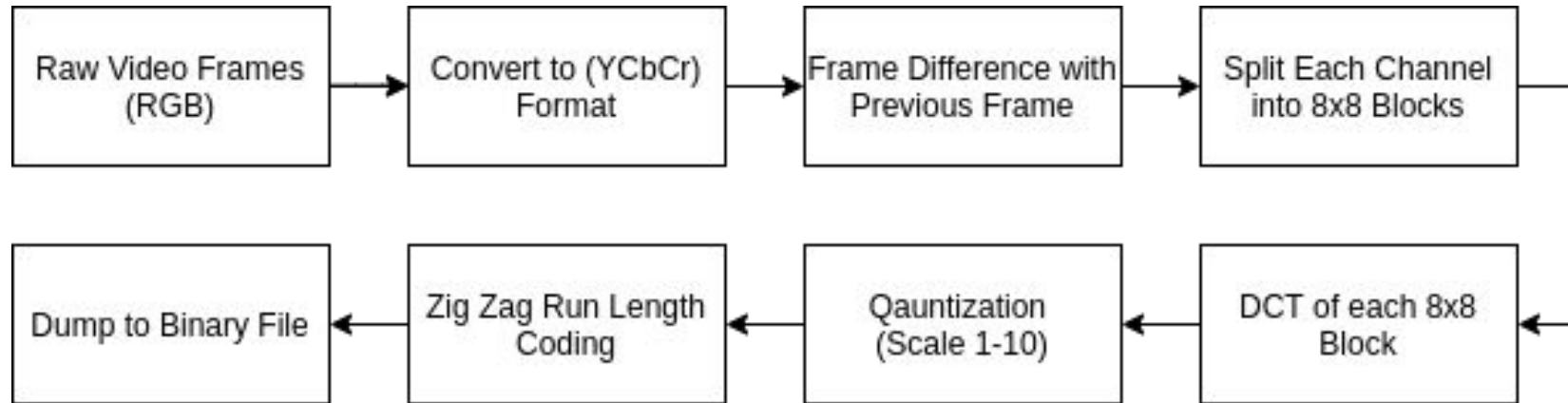
- **Pixel based** and **block based** are the two approaches used for motion estimation. In pixel based approach, the motion vectors are determined for every pixel in the image assuming brightness constancy. Due to various constraints, this approach requires large computation time.
- In block based approach, the candidate frame is divided into non-overlapping blocks, say of size 16x16 pixels, and a single motion vector is determined for the entire block.
- Block based motion estimation approach is easy to implement in hardware and real time motion estimation and prediction is possible. Hence, it is accepted in all video coding standards proposed till date.
- Different motion estimation algorithms have been proposed for block based approach. Full search block motion (FSBM) is the simplest but computational complex algorithm. Hence, several quick and efficient methods like 2-D logarithmic search, cross search algorithm (CSA), diamond search (DS) etc. have been proposed.

Simple Video Compression

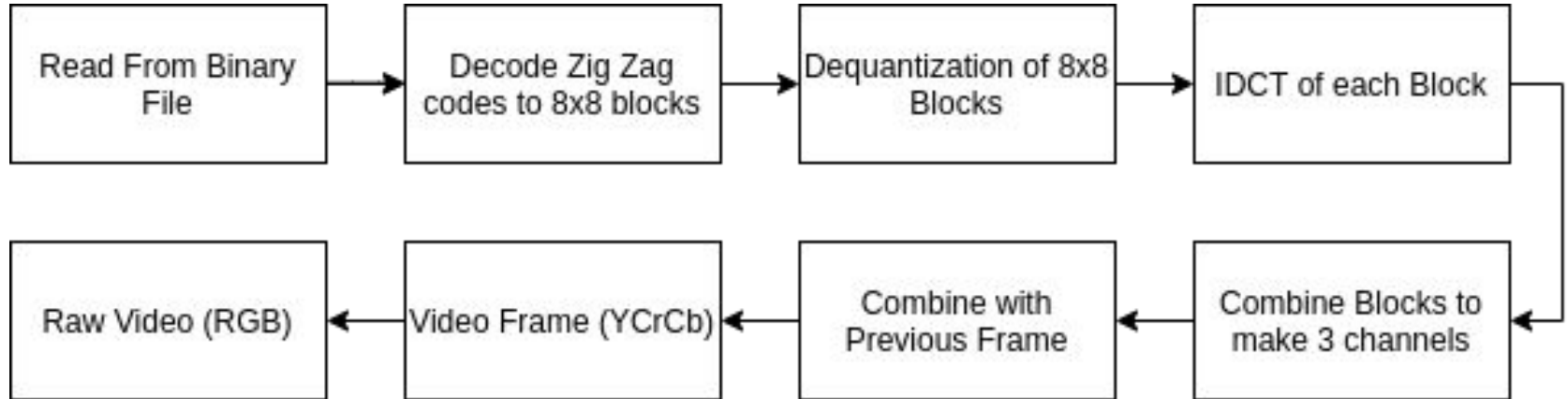
*Our own Video Compression Algorithmic Implementation using Python
and OpenCV*



Video Compression Flow



Video Decompression Flow



Let's do a Walkthrough

First 4x4 pixels of the image BGR format:

```
[[[ 18  21  18][ 17  20  17][ 16  20  21][ 17  21  23]]  
 [[ 18  21  18][ 18  21  18][ 16  20  21][ 16  20  21]]  
 [[  6   6   0][  3   3   0][  6   3   0][ 11   8   4]]  
 [[122 122 115][138 138 131][141 138 134][103 100  96]]]
```

First 4x4 pixels of the image YCC format:

```
[[[19.76 -0.76 -0.49], [18.76 -0.76 -0.49], [19.84 1.32 -1.67], [21.14 1.82 -1.84]]  
 [[19.76 -0.76 -0.49], [19.76 -0.76 -0.49], [19.84 1.32 -1.67], [19.84 1.32 -1.67]]  
 [[4.21 -2.5 1.51], [ 2.1 -1. 1.01], [2.45 -1.24 2.51], [7.15 -1.74 2.67]]  
 [[119.91 -3. 1.68], [135.91 -3. 1.68], [137.15 -1.74 2.67], [ 99.15 -1.74 2.67]]]
```

Let's do a Walkthrough

First 8x8 pixel's Luminance, The first block of the Image

```
[ [ 19.76  18.76  19.84  21.14  21.14  19.84  18.84  19.84 ]  
 [ 19.76  19.76  19.84  19.84  21.14  21.14  19.84  19.84 ]  
 [  4.21   2.1    2.45   7.15  11.4    5.99   0.59   0.59 ]  
 [119.91 135.91 137.15  99.15  89.4    122.29 130.13 137.13]  
 [236.39 255.    255.    191.76 133.81 192.4   250.94 255.   ]  
 [206.8   212.8   208.35 165.35 110.4   123.4   199.85 253.16]  
 [159.18 104.76 100.14 129.14 127.89 114.89 165.16 236.58]  
 [114.18  79.18  88.44 113.03 123.47 123.47 153.74 226.06] ]
```

Let's do a Walkthrough

DCT of this block

```
[[ 842.  -48.  119.  -65.   11.   12.    9.   -1.]  
 [-463.   70.  -93.   44.  -43.   -8.   -4.   -1.]  
 [-269.  -67.  -36.   35.   69.  -11.   -1.    4.]  
 [ 236.   43.   53.  -44.  -22.   15.   -6.   -1.]  
 [ 139.  -10.    1.   -0.  -31.    1.    9.   -4.]  
 [ -83.  -14.  -11.   19.   26.   -7.    0.    3.]  
 [ -70.   16.   -4.  -15.   -3.   -3.   -2.    1.]  
 [  -8.   -5.   -2.   12.    5.    9.   -2.   -1.]]
```

Let's do a Walkthrough

Quantization Matrix

```
[[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]]
```


After Quantization (Scale 1)

After Quantization (Scale 2)

```
[[ 53  -4  12  -4   0   0   0   0]  [[ 26  -2   6  -2   0   0   0   0]
[-39   6  -7   2  -2   0   0   0]  [-19   3  -3   1  -1   0   0   0]
[-19  -5  -2   1   2   0   0   0]  [-10  -3  -1   1   1   0   0   0]
[ 17   3   2  -2   0   0   0   0]  [  8   1   1  -1   0   0   0   0]
[  8   0   0   0   0   0   0   0]  [  4   0   0   0   0   0   0   0]
[ -3   0   0   0   0   0   0   0]  [-2   0   0   0   0   0   0   0]
[ -1   0   0   0   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]]
```

Run Length Coding

```
[(53, 1), (-4, 1), (-39, 1), ... , (0, 38)]
```

```
[(26, 1), (-2, 1), (-19, 1), ... , (0, 38)]
```


Function to compress a single channel

```
def single_channel_compression(img, scale_factor=1):  
    height = img.shape[0]  
    width = img.shape[1]  
  
    image_blocks = []  
    # split the image into 8x8 blocks  
    for j in range(0, height, 8):  
        for i in range(0, width, 8):  
            image_blocks.append(img[j:j+8, i:i+8])  
  
    # dct for each block  
    image_blocks_dct = [cv2.dct(image_block) for image_block in image_blocks]  
  
    # quantization of each block  
    reduced_blocks = [np.int8(np.round(dct_block / (QUANTIZATION_MATRIX * scale_factor)))  
                      for dct_block in image_blocks_dct]  
  
    return reduced_blocks
```

```

def compress_image(image, size, prev_image=None, scale_factor=1):
    img = np.float32(image)
    # Change to YCC color space
    img_ycc = cv2.cvtColor(img, cv2.COLOR_BGR2YCrCb)

    # User previous frame to reduce information
    if prev_image is not None:
        prev_image = np.float32(prev_image)
        prev_image = cv2.cvtColor(prev_image, cv2.COLOR_BGR2YCrCb)
        img_ycc = img_ycc - prev_image
    compressed_blocks = []

    for i in range(3):
        compressed = single_channel_compression(img_ycc[:, :, i],
                                                scale_factor)
        compressed_blocks = compressed_blocks + compressed

    # Convert 8x8 blocks into zigzag codes
    zigzag_pattern = []
    for block in compressed_blocks:
        zigzag_block = zig_zig_rlc(block)
        zigzag_pattern += zigzag_block

    zigzag_pattern = np.int8(zigzag_pattern)
    return zigzag_pattern

```

```
def compress_video(frames, size, file_name, scale_factor=1):

    complete_pattern = []
    frame_number = np.shape(frames)[0]
    for i in range(frame_number):
        print('Compression Frame', i)
        if i > 0:
            zigzag_pattern = compress_image(frames[i], size,
                                             frames[i-1], scale_factor)
        else:
            zigzag_pattern = compress_image(frames[i], size,
                                             None, scale_factor)

        complete_pattern = np.concatenate(
            (complete_pattern, zigzag_pattern))
        print(np.shape(complete_pattern))
        if i == 0:
            break

    complete_pattern = np.int8(complete_pattern)

    with open(file_name, 'wb') as f:
        pickle.dump(complete_pattern, f)
```

Comparison (4 x 4)



Uncompressed



Compressed

Comparison (8 x 8)



Uncompressed



Compressed

Comparison (16 x 16)



Uncompressed



Compressed

Comparison (32 x 32)



Uncompressed



Compressed

Comparison (64x64)



Uncompressed



Compressed

Results

Compression	Akiyo 352x288 300-Frames	Stefan 352x288 90-Frames
Raw Video	45.6 MB	13.7 MB
Simple Compression Scale 1	13.2 MB	8.1 MB
Simple Compression Scale 2	9.7 MB	6.0 MB
Simple Compression Scale 4	7.4 MB	4.3 MB
Simple Compression Scale 8	6.0 MB	3.0 MB
Simple Compression with Frame Difference	3.8 MB	7.4 MB
MPEG-4	651.6 KB	1.6 MB

Results (akioyo)



Raw



Scale 1



Scale 2

Results (akioyo)



Scale 4

Scale 8

Frame
Difference

Results (stefan)



Raw



Scale 1



Scale 2

Results (stefan)



Scale 4



Scale 8



Frame
Difference

MPEG-1 standard and its features

- MPEG-1 standard was primarily targeted for multimedia CD-ROM applications at a bit rate of 1.5 Mbits/sec. It specifies a syntax for representation of the encoded bitstream and the method of decoding with substantial flexibility in a number of parameters that can be specified in the bitstream itself including picture sizes and frame rates.
- The main features supported by MPEG-1 are:
 - Frame-based random access of videos
 - Fast-forward and fast reverse (FF/FR) searches
 - Reverse playback of video
 - Edit ability of compressed bit stream
- MPEG-1 works with (Y,Cb,Cr) colour space where Y is the luminance component and Cb & Cr are the two chrominance components.

Picture types in MPEG-1

1. Intraframe coded pictures (I-pictures)

- coded without reference to other pictures just like still images
- The first frame of every video sequence is an I-picture
- With no temporal prediction, they achieve very poor compression performance but allow random access and FF/FR functionalities in the bitstream.

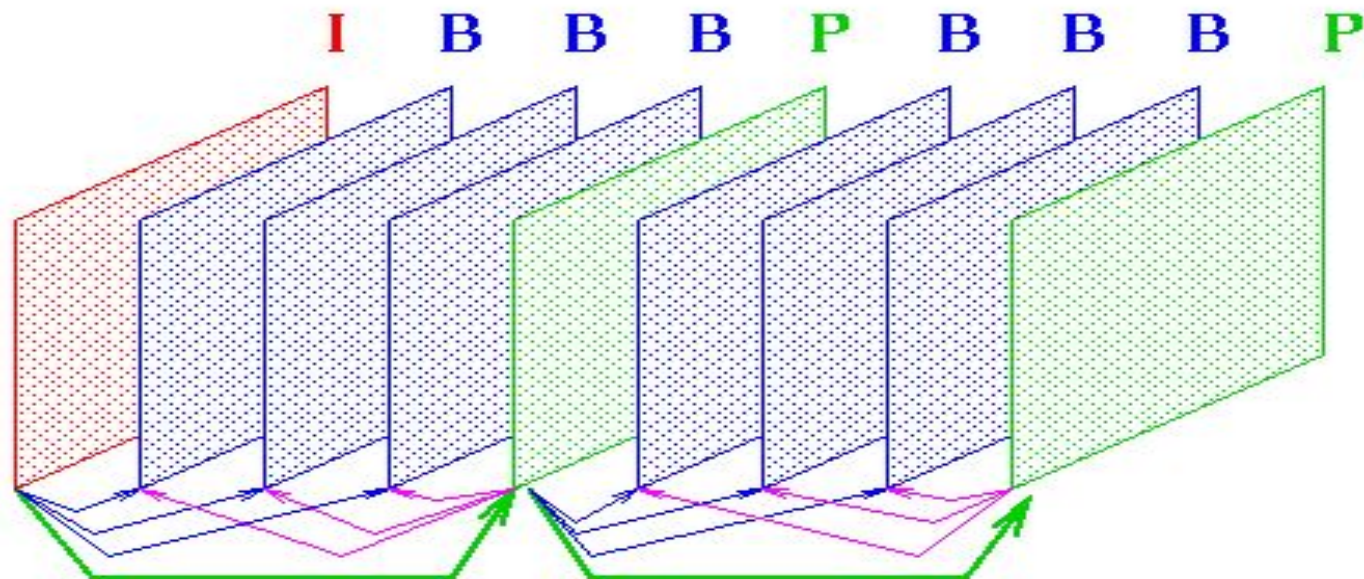
2. Interframe predicted pictures (P-pictures)

- coded with reference to the nearest coded I-picture or P-picture using motion compensation
- achieve better compression as compared to I-pictures but do not allow random access and FF/FR functionalities
- Temporal prediction does not work where there are scene changes.

3. Bi-directionally predicted pictures (B-pictures)

- Use bi-directional motion estimation with reference to the nearest coded I-picture and/ or P-picture on either side in temporal order
- Having best compression performance, almost two thirds of frames are B-pictures.
- also provides better SNR as compared to P-picture

MPEG display order



→ forward prediction of P-frames

→ forward prediction of B-frames

→ backward prediction of B-frames

Hierarchical data structure in MPEG-1

- MPEG-1 datastream follows a 6 layer hierarchical structure.
- The video sequence is divided in several group of pictures (**GOPs**).
- Each of the six layers have headers to uniquely specify the data that follows.

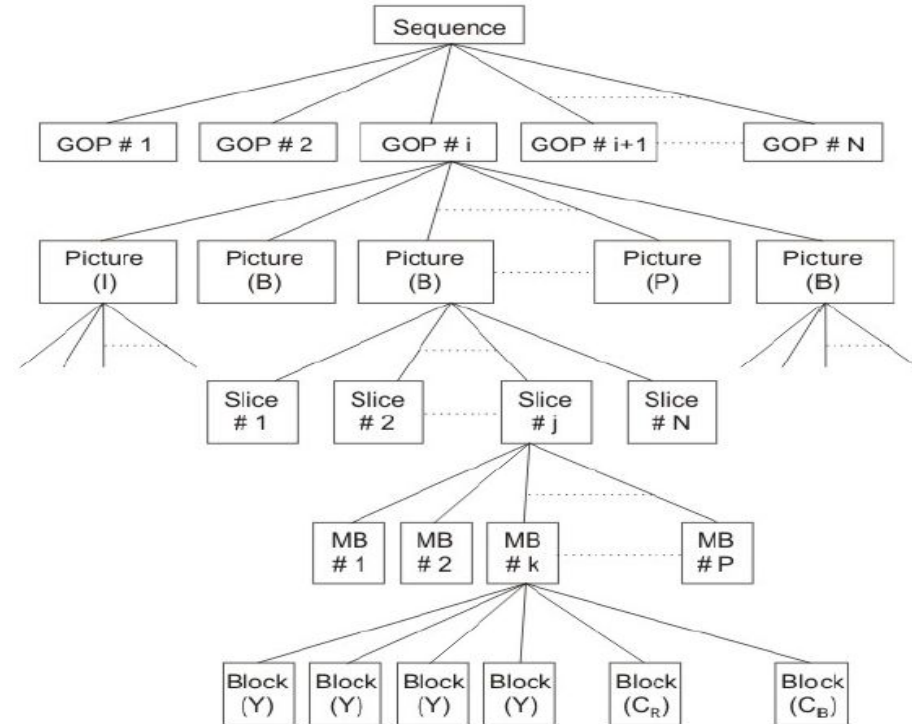
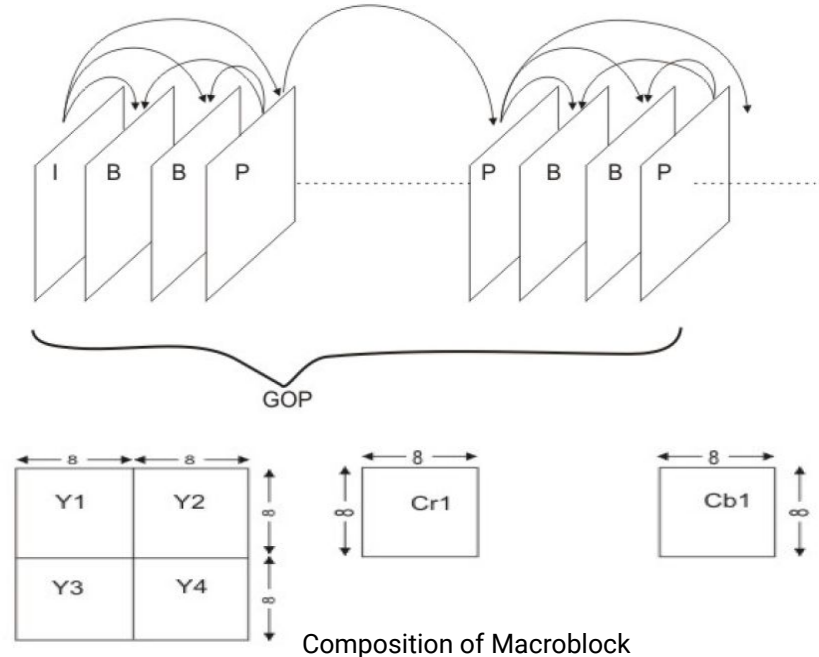


Image Source : NPTEL

Hierarchical data structure in MPEG-1

- Each GOP begins with an I - picture.
- P-pictures are encoded at regular intervals (3 or 4 frames) and the frames between the I-picture and P-pictures are encoded as B-pictures.
- The pictures are composed of slices which are sequence of **macroblocks** (MBs).
- Each macroblock is composed of 16X16 pixels of Y channel (subdivided into four blocks of 8X8 pixels) and one block of 8X8 pixels from each Cr and Cb channels. Hence, each macroblock consists of six 8X8 blocks.



Intra frame encoding for I-pictures

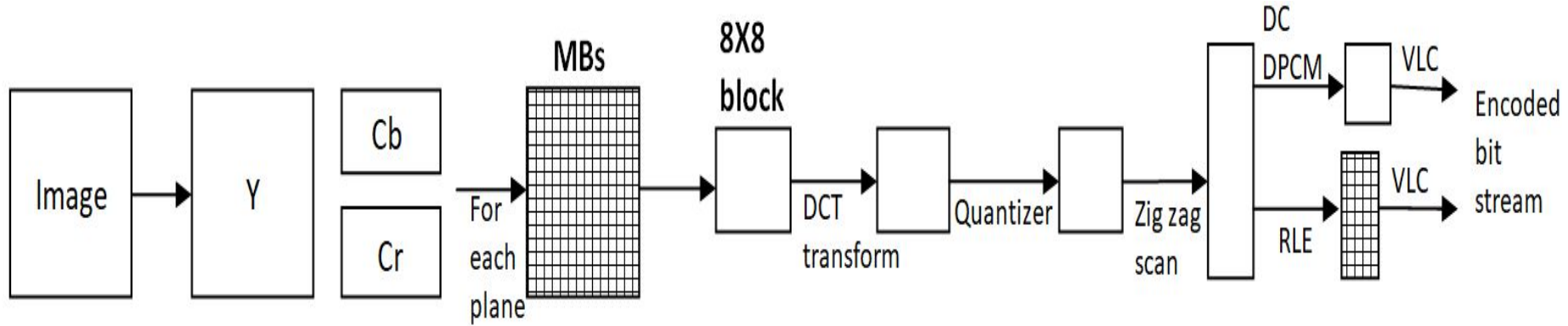
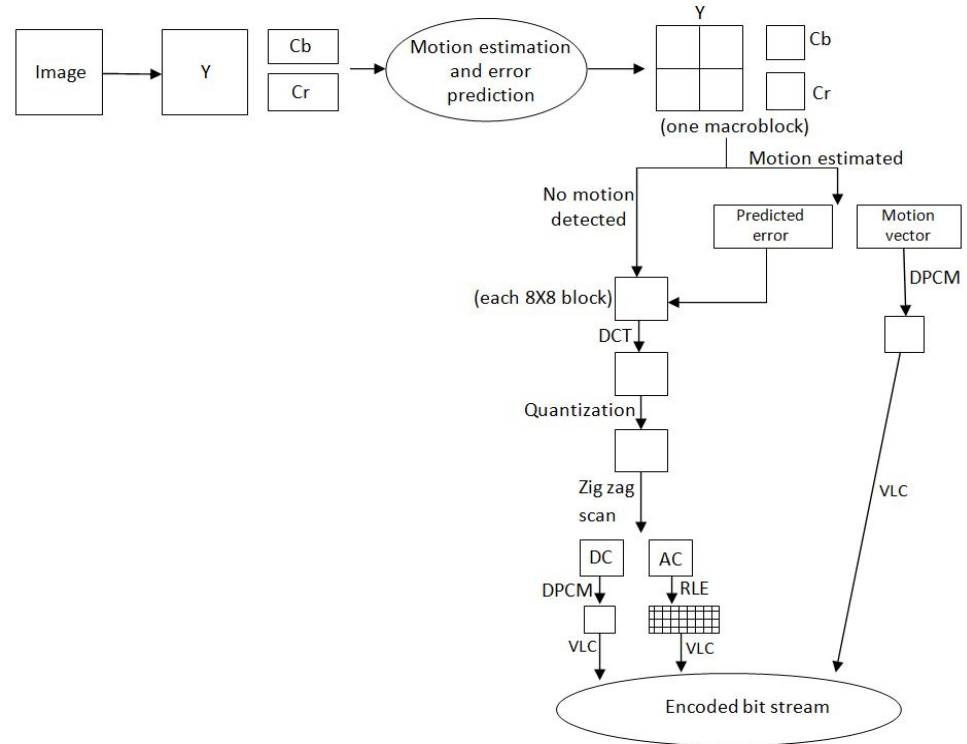


Image Source : www.cmlab.csie.ntu.edu.tw

- A total of six 8X8 blocks are encoded for each macroblock.
- DPCM: Difference pulse coded modulation (predictive coding) used to encode the DC coefficient only.
- RLE: Run length encoding used to encode the AC coefficients
- VLC: Variable length coding used to encode the DPCM and RLE encoded coefficients

Interframe encoding for P-pictures and B-pictures

- In case of P-picture & B-pictures, motion estimation is done using a reference frame from the frame buffer. The difference in current frame and motion compensated frame is predicted (error).
- If, there is no error, the frame is encoded just like an I-picture. If a motion error is present, the error is encoded using DCT transform and the motion vector is encoded using DPCM and added to the bitstream.
- The frame is reconstructed from the DC and AC coefficients after quantization and stored in the frame buffer if required.



MPEG-2 standard

- MPEG-2 was developed as a standard for high quality multi-channel, multimedia signals for terrestrial broadcast, digital and cable TV distribution etc. It was also adopted in DVDs.
- It was designed for coding interlaced images with bit rate around 4 Mbps. In **interlaced scanning**, the frame is partitioned into odd field (set of odd-numbered scan lines) and even field (set of even-numbered scan lines). Each field is coded separately which gives better results in presence of significant motion. Such pictures are known as **field pictures**.
- Various **profiles and levels** are defined to suit wide range of applications. Each profile adds a new set of algorithms and a level specifies the range of parameters like image size, frame rates etc.
- It supports **scalability** to provide interoperability between different services and to support receivers with different display capabilities. The bit stream is organized into two or three layers. The bottom layer is the base layer which every receiver must make use of above which are the enhancement layers for high end applications. This finds use in SDTV and HDTV applications.
- Every MPEG-2 compatible decoder can decode a valid MPEG-1 bit stream i.e. backward compatibility with MPEG-1.

MPEG-4 standard

- Unlike MPEG-1 & MPEG-2, MPEG-4 was not confined to rectangular sized pictures. It adopted an **object based coding** concept to individually encode and transmit arbitrarily shaped and dynamically changing individual audio-visual objects in a video sequence.
- It addressed very low bit rate coding (5 - 64 Kbits/sec) to 2 Mbits/sec for TV/film applications.
- It has a wide range of applications including internet streaming, wireless video, digital video cameras, mobile phones etc.
- It has an ability to **encode multiple views**, for example stereoscopic video.
- An important feature of MPEG-4 is **sprite coding**. Sprite assumes a flat and static background. Using video segmentation algorithms to extract the foreground and background, the background is encoded separately and only transmitted once before the foreground.

H.264 standard

The main features of H.264 standard are as follows:

- Improved motion estimation upto quarter- pixel accuracy
- Use of **4X4 integer transforms** in place of 8X8 DCT
- Advanced prediction modes for intracoded and inter-coded frames
- Improved context based arithmetic entropy encoding
- Enhanced compression performance in conversational (like video telephony and video conferencing) and non conversational (like storage, broadcast and streaming) applications
- A **wavelet based filter** to reduce the artefacts caused because of separate compression of individual macroblocks

Comparison and Performance Metrics???

- Video Codecs cannot be compared based on just the “*Compression Ratio*” or “*Encoding/Decoding Time*”
- Resultant **Video Quality** and **Compression Artifacts** are major factor which cannot be easily quantified
- Industries use various codecs depending on their use cases

Future of Video Compression?

- The most popular video coding standards used for codecs have been the MPEG standards.
- The most widely used video coding format, as of 2016, is H.264/MPEG-4 AVC (2003).
- H.264 is the used in Blu-ray Discs, YouTube, Netflix, Vimeo, iTunes Store, Adobe Flash Player and HDTV television broadcasts.
- AVC has been succeeded by HEVC/H.265 (2013), it is currently heavily patented.
- MPEG also working on many future standards like VVC/H.266, MPEG-5 Part 1/EVC and MPEG-5 Part 2/LCEVC, all of which are expected to release in 2020/2021

Ever Changing Applications

Applications and Requirements of Video Codecs are ever changing with new uses and each needed their special codecs

- Video Calling
- Live Streaming (Sports and TV)
- Sensor data
- Game Streaming
- 180° and 360° Video
- Virtual Reality Content

Thank You !!!!

All Question and Queries to:
Himali Singh (160003020)
Khushboo Ahuja(160002024)
Naman Singhal(160001038)

References

- *Raid, A.M., Khedr, W.M., El-Dosuky, M.A. and Ahmed, W., 2014. Jpeg image compression using discrete cosine transform-A survey. arXiv preprint arXiv:1405.6147.*
- CMLAB - Communications and Multimedia Laboratory
(cmlab.csie.ntu.edu.tw/cml/dsp/training/coding/mpeg1/index.html)
- NPTEL - National Programme on Technology Enhanced Learning (nptel.ac.in/courses/117105083/)
- <https://web.stanford.edu/class/ee398a/handouts/lectures/01-EntropyLosslessCoding.pdf>
- Raw Video Data - <http://trace.eas.asu.edu/yuv/>
- Wikipedia.org
- ScienceDirect (<https://www.sciencedirect.com/topics/engineering/uniform-quantization>)