Daffodil International University

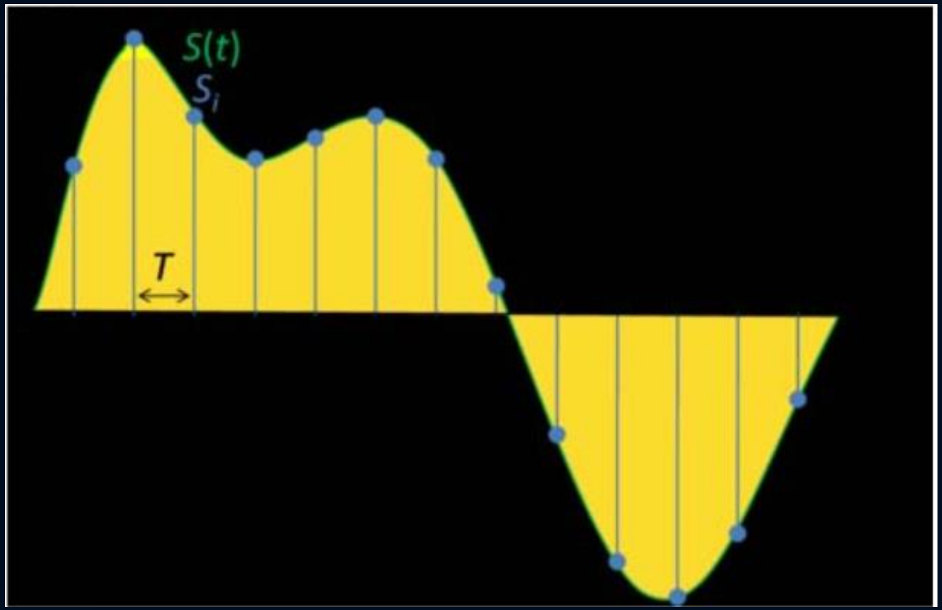
Broadcast Technologies
ETE 457

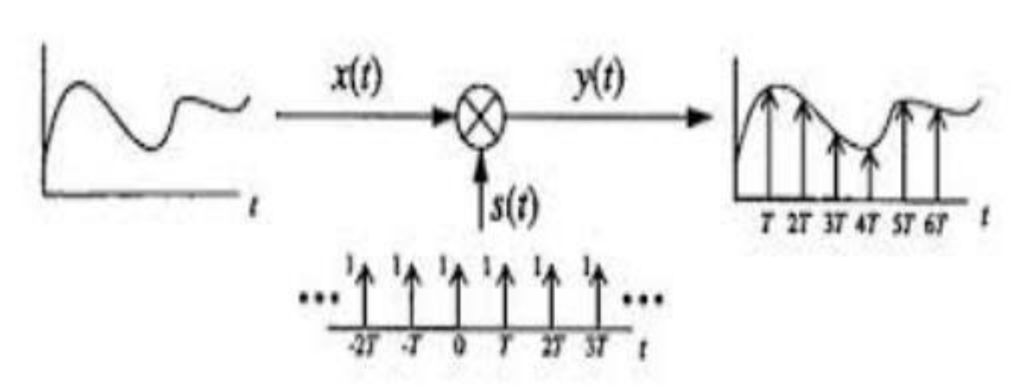
LECTURE 2
VIDEO COMPRESSION: BASIC

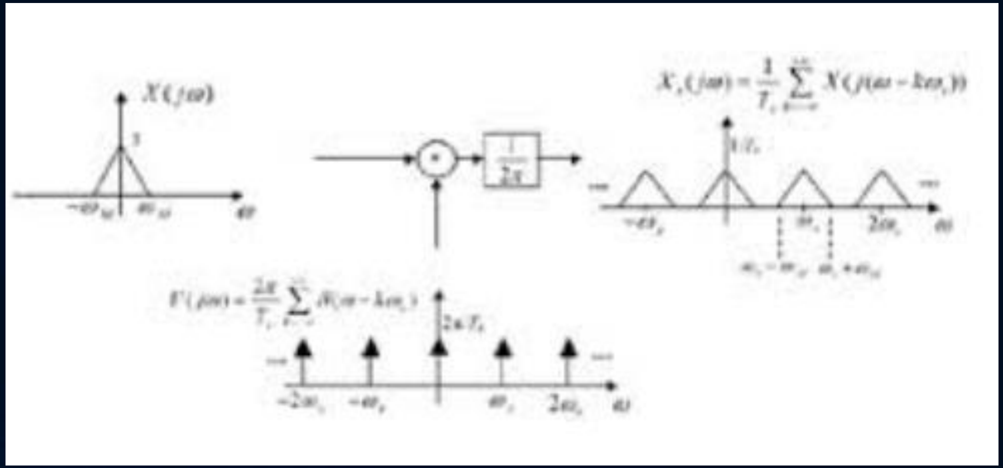
Lecture Content

- In this lecture, we will cover:
  - Sampling.
  - Quantization.
  - Digital Interfaces: SDI, ASI, etc.
  - Picture Compression: JPEG.
  - Moving Picture Compression: MPEG.

Sampling

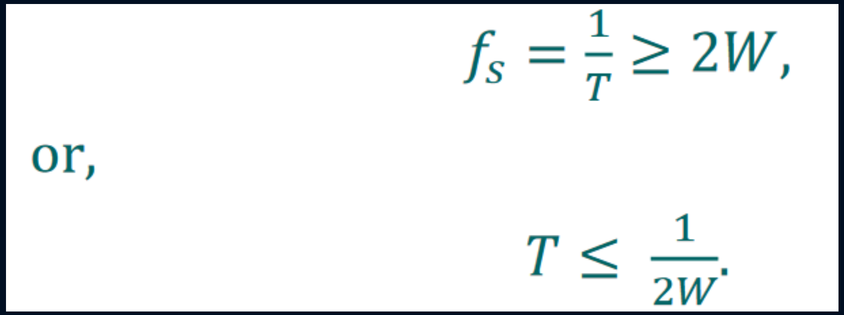

$S(t)$
S_i
T


t
$x(t)$
$y(t)$
$s(t)$
T $2T$ $3T$ $4T$ $5T$ $6T$
t
1 1 1 1 1 1
$-2T$ $-T$ 0 T $2T$ $3T$
t


$X(j\omega)$
1
$-\omega_M$ ω_M
ω
$\frac{1}{2\pi}$
$X_s(j\omega)=\frac{1}{T_s}\sum_{k=-\infty}^{+\infty}X(j(\omega-k\omega_s))$
$1/T_s$
$-\omega_s$ ω_s $2\omega_s$ ω
$\omega_s-\omega_M$ $\omega_s+\omega_M$
$P(j\omega)=\frac{2\pi}{T_s}\sum_{k=-\infty}^{+\infty}\delta(\omega-k\omega_s)$
$2\pi/T_s$
$-2\omega_s$ $-\omega_s$ ω_s $2\omega_s$ ω

Sampling

- Nyquist theorem:
  - The number of samples per second, i.e., the sampling rate should be more than or equal to twice the highest frequency of the analog signal:



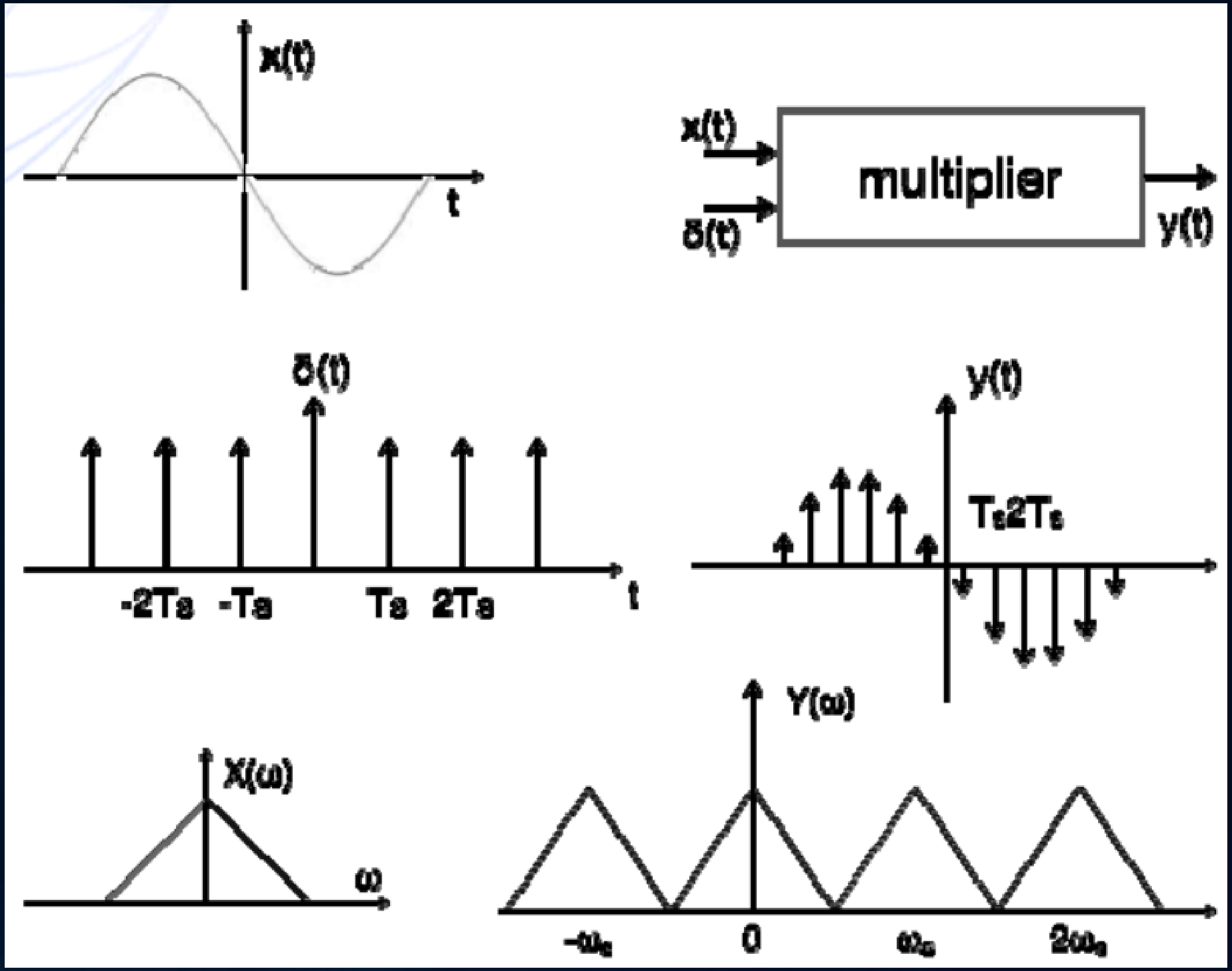
$$f_s = \frac{1}{T} \geq 2W,$$

or,

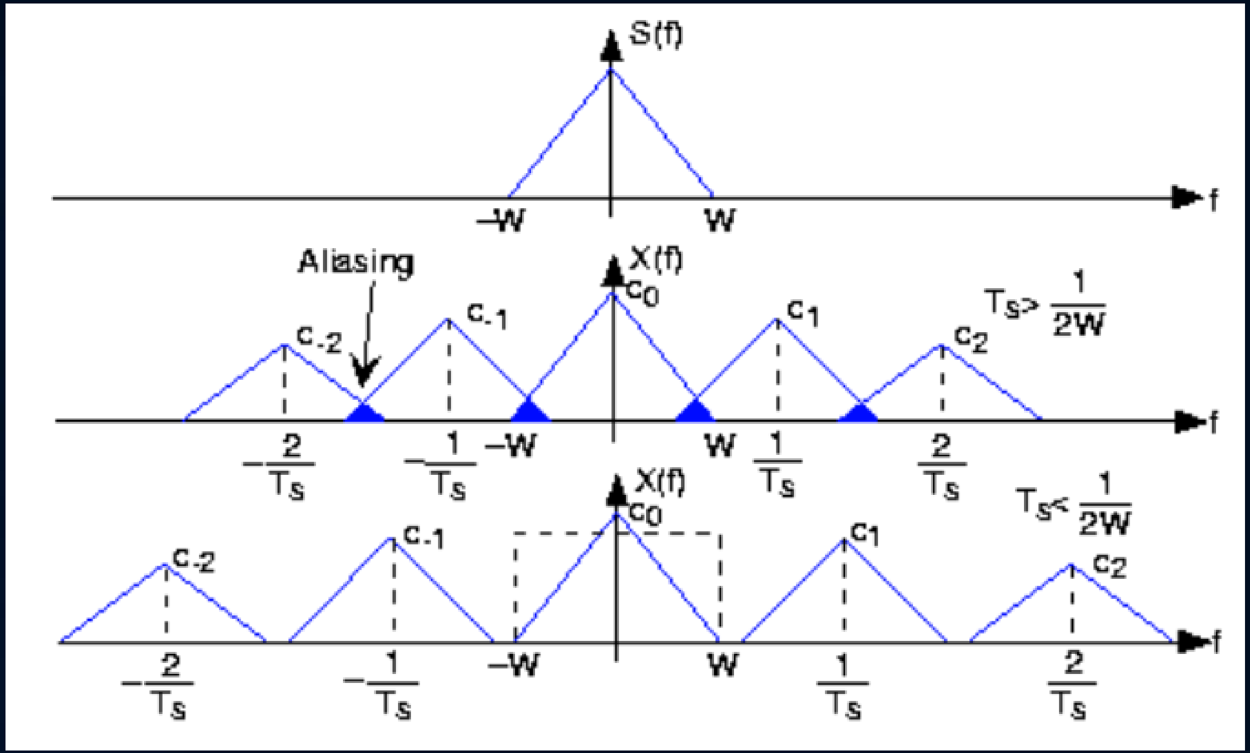
$$T \leq \frac{1}{2W}.$$

- If $T > \frac{1}{2W}$ there will be aliasing.

Critical Sampling



Aliasing


S(f)
f
–W
W
Aliasing
X(f)
c_0
c_{-1}
c_{-2}
c_1
c_2
$T_s > \frac{1}{2W}$
$-\frac{2}{T_s}$
$-\frac{1}{T_s}$
$\frac{1}{T_s}$
$\frac{2}{T_s}$
$T_s < \frac{1}{2W}$

Sampling: Voice and Audio

- Human voice has frequencies up to 3.4 kHz. So, the minimum sampling frequency for voice is 6.8 k samples/sec. But to make filtering easier a sampling rate of 8 ksps is used, i.e., one sample every 125 µ seconds.
- For an audio signal a frequency range of 20 to 20,000 Hz. is considered. This is the range of frequencies human auditory system can detect. So, at least 40,000 samples per second is required. Usually a sampling rate of 44.1 ksps is used for audio.

Sampling: Image

While sound is a one dimensional, temporal (time varying) signal, an image is a two-dimensional spatial signal. Therefore, the sampling interval instead of having dimension of time, has the dimension of length. The horizontal and vertical resolution is defined in term of ∇x and ∇y, respectively or their inverse n_x and n_y. They represent the number of samples (pixels) in each row and column. The number of pixels depends on the frequency in X and Y dimension. A “busy” image has higher frequency components hence needing more samples.

Sampling: Video

Video or moving picture is a sequence of images in time, so in addition to two spatial dimensions it is also a function of time, $f(x, y, t)$. The number of pixels in a frame determines the spatial resolution. The temporal resolution is determined by the number of images shown (frames) per second.

In the previous lecture, we saw that in order to prevent flickering and at the same time not to increase the data rate, a frame sometimes is divided into two fields (odd and even fields) and fields are shown in an interlaced fashion. This is called interlaced scan as opposed to progressive scan where the whole frame is scanned. For example 480p denotes a format with 480 lines per frame and $480 \times \frac{4}{3}$ = 640 pixels per line if the aspect ratio is 4/3 and 853 pixels per line for the aspect ratio of 16/9.

Sampling: Video

- Similarly 720p (HD) is a video format with 1280 x 720 pixels per frame.
- 1080i (HD) and 1080p (True HD) refers to a 1920 x 1080 pixel interlaced or progressive scan format, respectively.
- 2160p or 4K format with 3840 x 2160 pixels called UHDTV.

The number of frames per second can be 25p, 30p, 50i, 60i, 50p/60p, 100p/120p, 300p

Analog to Digital Conversion

- After sampling a source whether audio or video, we need to convert the voltage level obtained into a finite number of values so that we can represent each sample of the audio signal or each pixel with a finite number of bits.

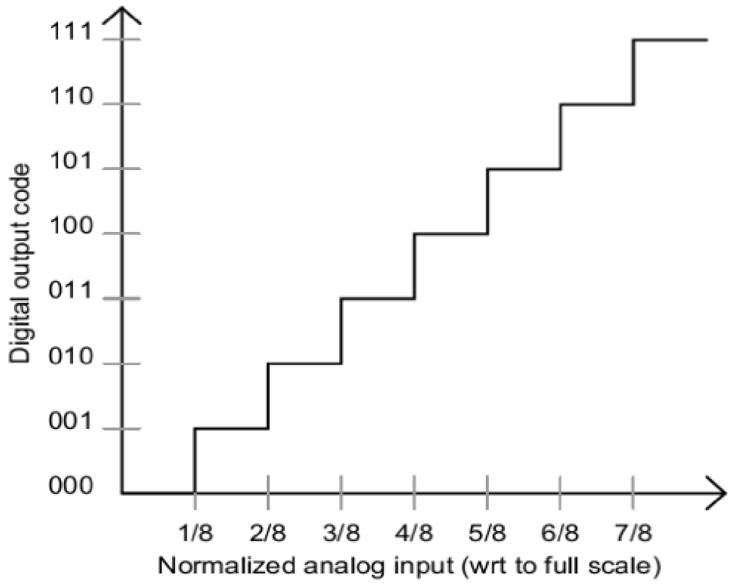


Fig : Uniform Quntization

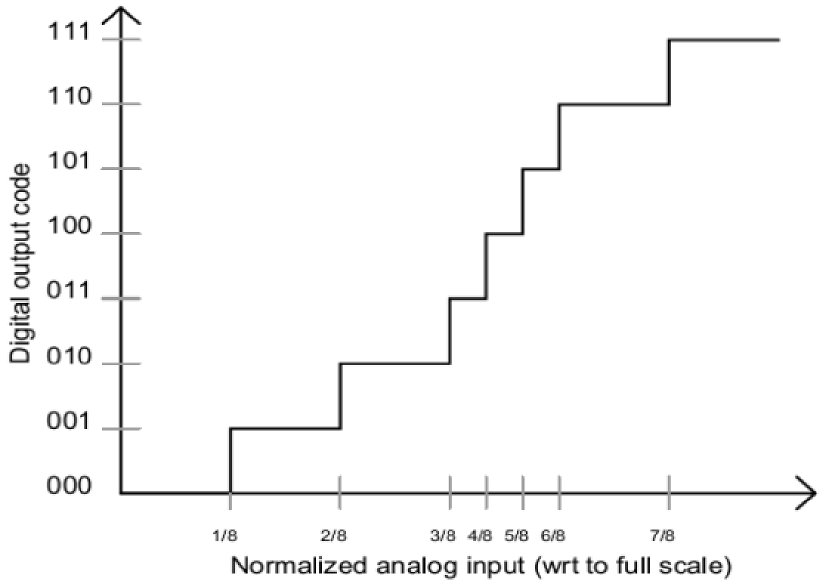


Fig : Non-uniform Quntization

Raw bit rate

We so far have learnt about the number of samples required to represent a source whether temporal or spatial (pixels in image or video) and the fact that each bit added to the ADC gives us roughly an extra 6 dB of signal quality. Now let’s find what is the bit rate for transmitting or storing a sampled and quantized source. We do this for voice, audio and video. This, particularly in the case of video, gives ridiculously large values. This gives us an appreciation for the work gone into audio and video compression as well as advanced digital coding and modulation techniques that have brought the bit rates into a reasonably low value allowing us to receive and retrieve audio and video signal with extremely high quality over a large variety of platforms.

Raw bit rate

- Let’s start with voice signal. We said that voice signals can be represented by samples taken every 125 µs .That is they are sampled at the rate of 8000 samples per second. If we are content with 48 dB of SQNR, we can represent each sample with 8 bits. This means that in order to transmit the voice signals over the phone, we need 64 kbps. This is actually the rate used at the start of digital telephony. It was called a Voice Channel (VC). The technique was called Pulse Code Modulation (PCM). With the advances in voice coding rate were reduced to 32 kbps (DPCM), 16 kbps (ADPCM) and less than 8 kbps with Linear Predictive Coding (LPC) techniques such as CELP.

- For audio sampling is at the rate of 44.1 ksps and each sample is quantized with 16 bits so the total rate is 2 x 44.1 x 16 ≈ 1.411 Mbits/sec. This is called CD format. Using mp3 roughly the same perceptual can be obtained with 128 kbps (1/11 compression).

Raw bit rate

- For Video, we need three color samples per pixel. So, if the number of pixels are n_x and n_y and there are n_f, the raw bit rate will be

$$R_b = 3n_b n_f n_x n_y$$

- Let's take 1080i Then,

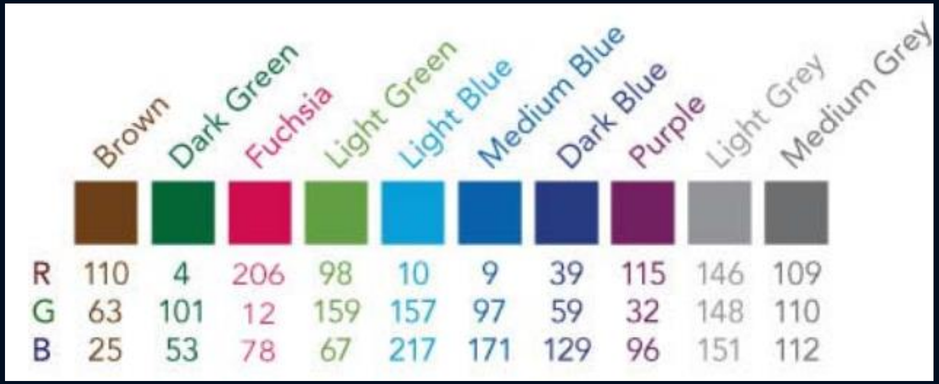
$$R_b = 3 \times 8 \times 30 \times 1080 \times 1920 \approx 1.5 \text{ Gbps}$$

In the above, we have assumed 8 bit quantization. The industry is moving towards 10 bit for some applications.

- A Blu-ray disk that has 50 GB capacity can only store 4.5 minutes of raw video. And we have not even added audio and metadata. So, let's see what compression can do for us.

RGB

- Any color can be represented as a linear combination of Red Green Blue

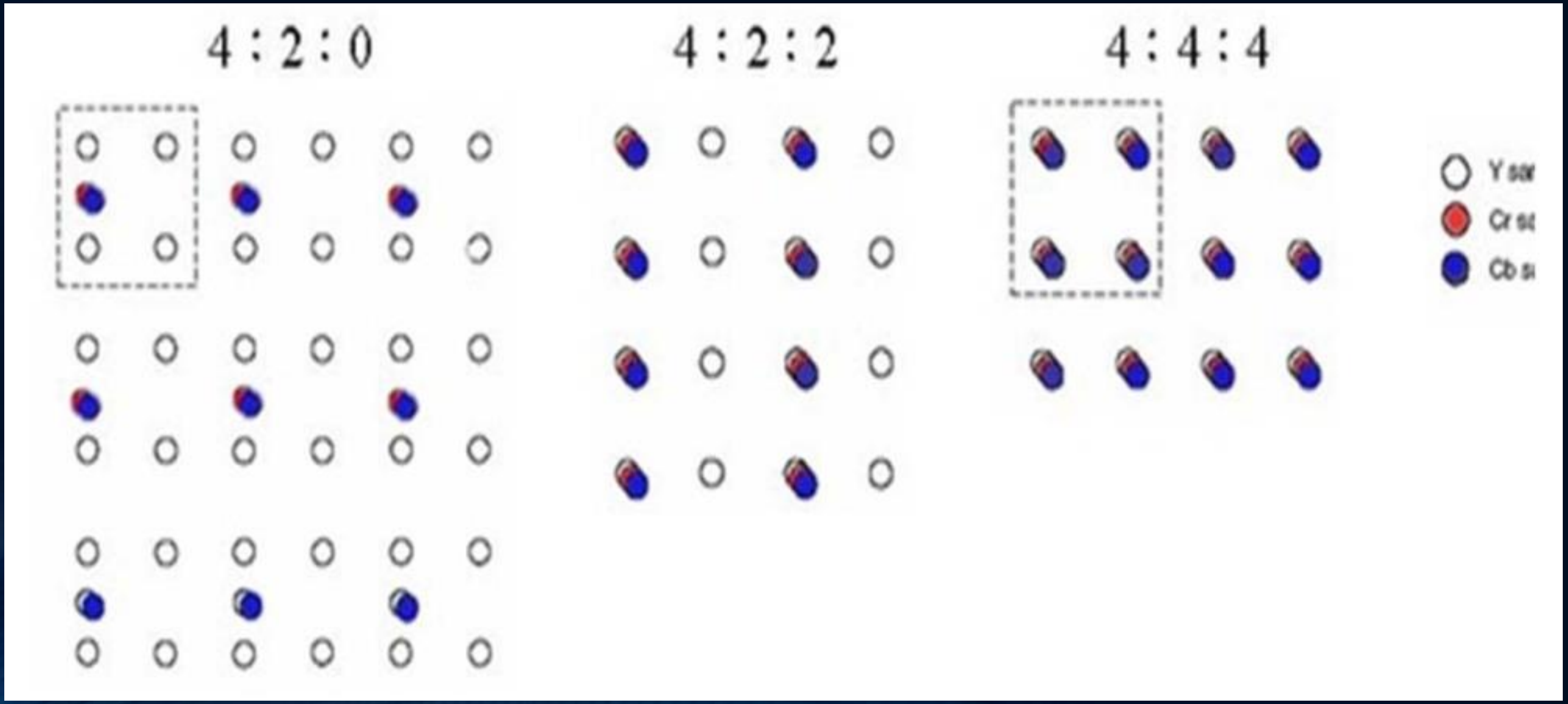


	Brown	Dark Green	Fuchsia	Light Green	Light Blue	Medium Blue	Dark Blue	Purple	Light Grey	Medium Grey
	🟫	🟩	🟥	🟩	🟦	🟦	🟦	🟪	⬜	⬛
R	110	4	206	98	10	9	39	115	146	109
G	63	101	12	159	157	97	59	32	148	110
B	25	53	78	67	217	171	129	96	151	112

- As we saw in the previous lecture the three color scheme RGB or Component can be transformed into YC_bC_r or sometimes called YUV or Lab, where the first component is the luminance and the other two are Chroma. The simplest way to transform component into composite is Y=R+G+B, C_b=Y-B and C_r=Y-R . However, usually some matrices are used based on the human visual perception of color.

RGB

- The first thing we do to reduce the image (or video) size is to use the fact that the eye is less sensitive to color than to brightness. So instead of having two colors for each Y (4:4:4), we can use two colors for each 4 Y's (4:2:0) or 4 colors for each Y (4:2:2).



SDI Interface

- Serial Digital Interface is a standard developed by The Society of Motion Picture and Television Engineers (SMPTE) in 1989 (SMPTE 259M). It is used for transferring uncompressed video. It uses BNC connector. Later high-definition SDI (HD-SDI), was standardized in SMPTE 292M; this provides a nominal data rate of 1.5 (1.485) Gbit/s. It is good for transferring a single 4:2:2 video. Later 3G-SDI was developed that can carry a 1080p 50/60 or 1080p 4:4:4.

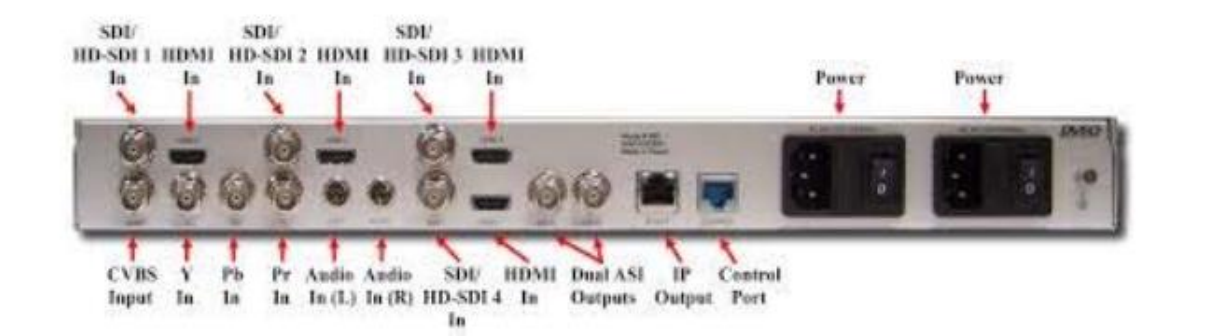
6G-SDI and 12G-SDI have 6 and 12 Gb/s rate, respectively and can be used to transport 4K video.

HDMI Interface

- HDMI implements the EIA/CEA-861 standards, which define video formats and waveforms, transport of compressed, uncompressed, and LPCM audio and auxiliary data. HDMI 2.0 released in 2013 has the maximum bitrate of 6 Gbit/s for each channel and a total throughput of 18 Gbit/s. This allows HDMI 2.0 to carry 4K resolution at 60 frames per second (fps).

ASI

- While SD-SDI (270 Mbit/s) or HD-SDI (1.485 Gbit/s) carry uncompressed video an, an ASI (Asynchronous Serial Interface) signal can carry one or multiple SD, HD or audio programs that are already compressed. ASI, is a streaming data format which often carries an MPEG Transport Stream (MPEG-TS). ASI signal can be at varying transmission speeds and is completely dependent on the user's engineering requirements. For example, an ATSC has a maximum bandwidth of 19.392658 Mbit/s. Generally, the ASI signal is the final product of video compression, either MPEG2 or MPEG4, ready for transmission to a transmitter or microwave system or other device. Sometimes it is also converted to fiber, RF or SMPTE310 for other types of transmission. There are two transmission formats commonly used by the ASI interface: the 188 byte format and the 204 byte format. The 188 byte format is the more common ASI transport stream. When optional Reed–Solomon error correction data are included, the packet can stretch an extra 16 bytes to 204 bytes total.



JPEG

- The goal of video compression techniques is to reduce the bit rate of the video while keeping the quality as high as possible. The reduction in bit rate (size of the video file) is achieved by removing the redundancy in the video signal. Being a three dimensional signal, a video contains spatial and temporal information and hence spatial and temporal redundancy. The spatial redundancy is the redundant content in a frame (intra-frame redundancy) while the temporal redundancy is the similarity between the consecutive frames (inter-frame redundancy). A frame full of different objects of varying size and color has hardly any redundancy and cannot be compressed a lot while a quiet frame has lots of redundancy and can be compressed considerably. Similarly a slowly varying scene, say, a broadcaster reading the news has hardly any inter-frame variation and there is a lot of redundancy to be removed hence high compression ratio. On the other hand in a football match there is a lot changing between two frames and therefore, we need more bits to represent the video.

JPEG

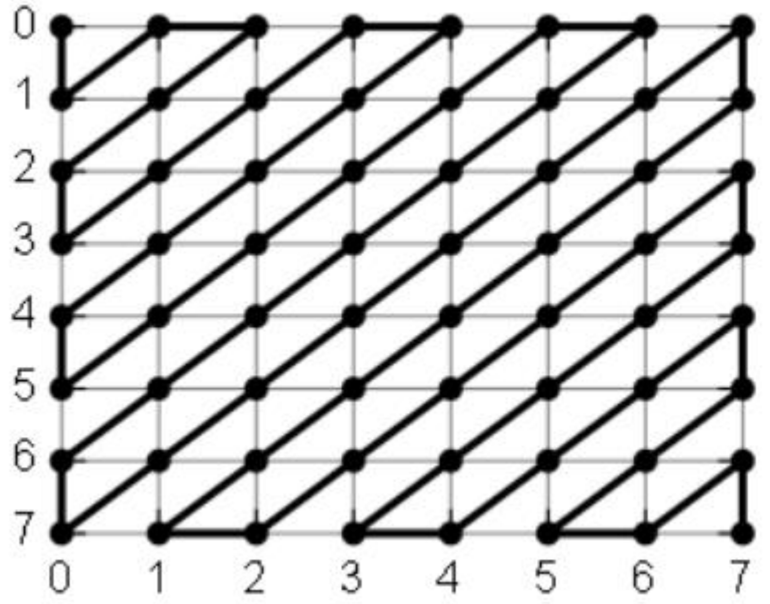
- Removing the spatial redundancy is done using transform coding (in order to find the prominent spectral lines of a frame) and some statistical techniques such as Huffman Coding in order to assign less bits to more probable values.
- Removal of the temporal redundancy is through utilization of the similarity between consecutive frames by sending information in the form of motion vectors just enough for the decoder to be able to reconstruct the frame based on the older (fresh or reconstructed) frames and the incremental data contained in the motion vectors.
- We start the discussion with the spatial compression, i.e., encoding a frame and will later discuss about inter-frame compression. As a frame is just a picture, we start with image compression technique JPEG which is basically the same technique used in different versions of MPEG.

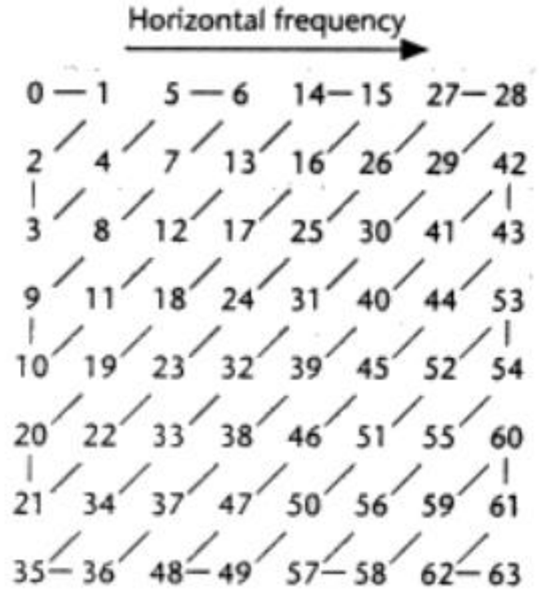
JPEG

- JPEG was developed by the Joint Photographic Experts Group.
- JPEG uses a lossy compression technique based on the Discrete Cosine Transform (DCT). DCT converts each frame/field of the video source from the spatial (2D) domain into the frequency (transform) domain A perceptual model based loosely on the human psychovisual system discards high-frequency information, i.e. sharp transitions in intensity, and colour hue. In the transform domain the picture is quantized by discarding or highly compressing the high frequency coefficients, which contribute less to the overall picture than other coefficients and are also characteristically small-values with high compressibility. The quantized coefficients are then sequenced and losslessly packed into the output bitstream.
- Lossless coding of transform domain coefficients is done using Huffman coding.

JPEG: Zigzag scan

- In JPEG, in order to preserve correlation between pixels, zigzag scan is used on an 8-by-8 group of pixels:





0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

JPEG: Huffman Coding

- The average length of a data stream consisting of a characters taking value from a given alphabet can be reduced by assigning shorter representation (less bits) to more frequently appearing characters. For example in compressing a written text, say in English, instead of assigning the same number of bits to each letter, we may assign less bits to more frequent letters such as t or e and more bits to q or z.

- In order for a code to be useful, it has to be prefix-free, i.e., no codeword can be prefix for another. These are called prefix codes.

- The optimum prefix codes are designed based on the following rules:

- For a source with alphabet having probabilities $p_1, p_2 \ldots$

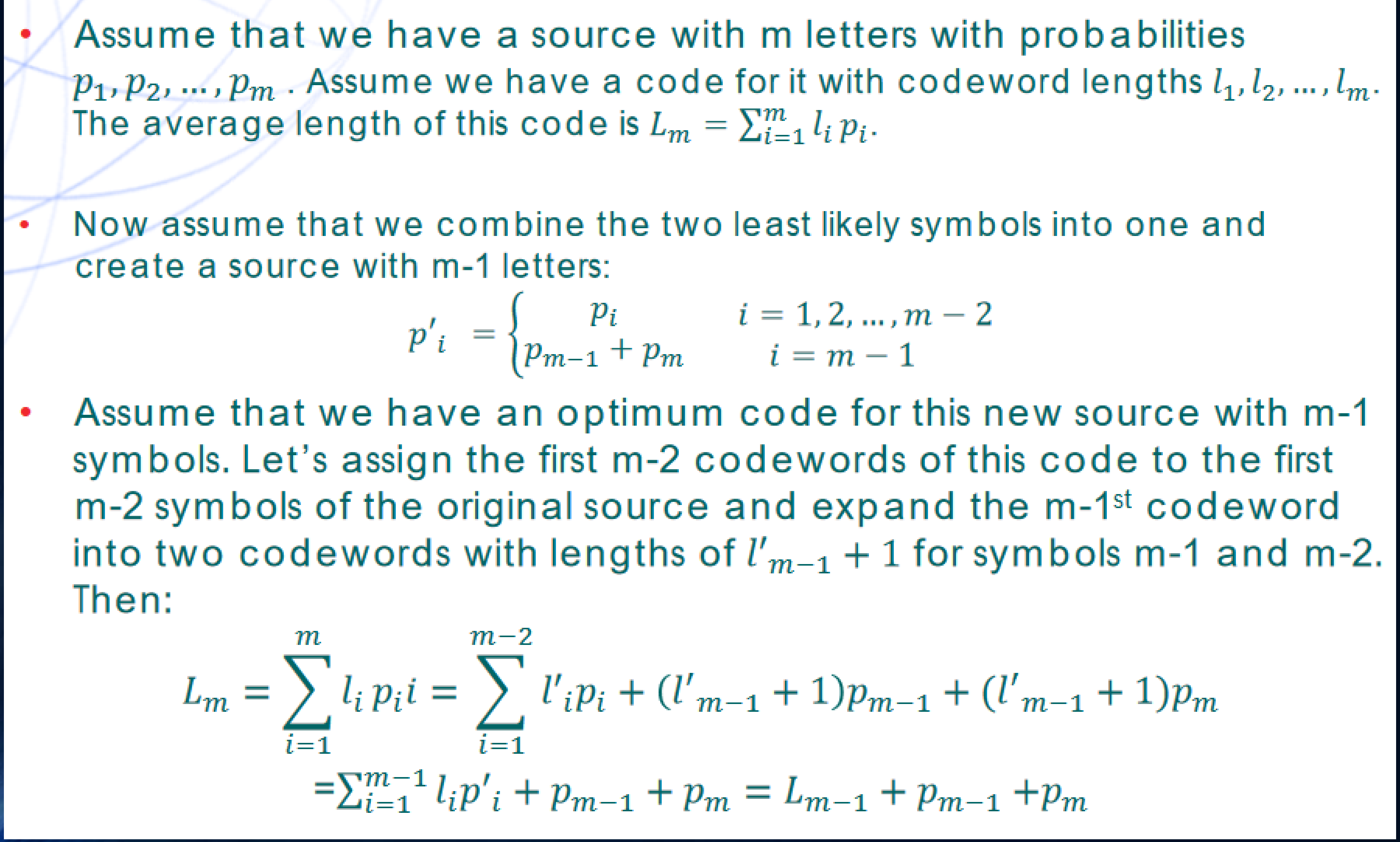
  – If $p_j > p_k$ then $l_j \leq l_k$ l_j and l_k are the lengths of symbols j and k.

  – The two longest codewords (corresponding to the least probable symbols) have the same length.

  – The two longest codewords differ only in the last bit.

- These rules provides us a tool for designing optimal (Huffman) Codes

JPEG: Huffman Coding



- Assume that we have a source with m letters with probabilities $p_1, p_2, \dots, p_m$. Assume we have a code for it with codeword lengths $l_1, l_2, \dots, l_m$. The average length of this code is $L_m = \sum_{i=1}^{m} l_i \, p_i$.
- Now assume that we combine the two least likely symbols into one and create a source with m-1 letters:

$$p'_i \ = \begin{cases} p_i & i = 1, 2, \dots, m-2 \\ p_{m-1} + p_m & i = m-1 \end{cases}$$

- Assume that we have an optimum code for this new source with m-1 symbols. Let's assign the first m-2 codewords of this code to the first m-2 symbols of the original source and expand the m-1st codeword into two codewords with lengths of $l'_{m-1} + 1$ for symbols m-1 and m-2. Then:

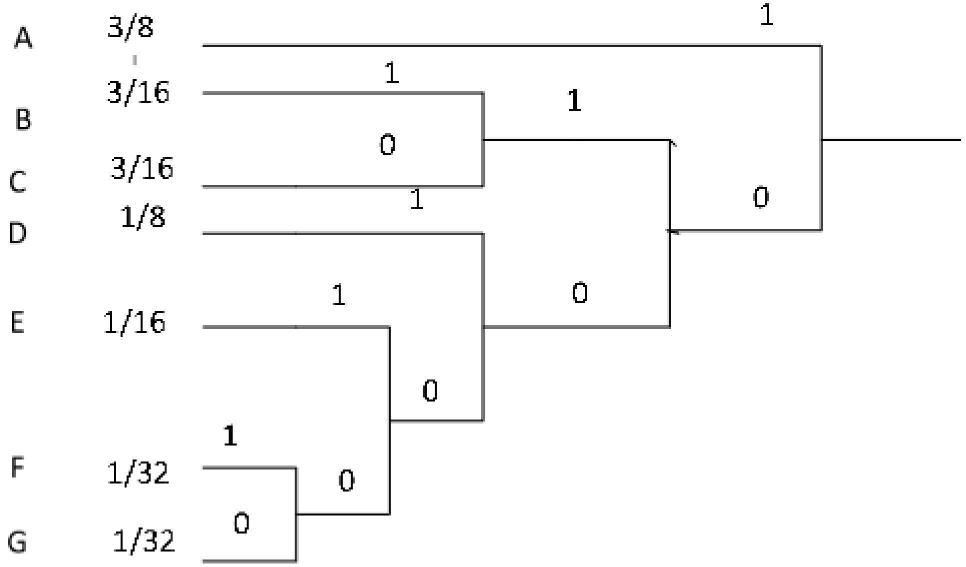
$$L_m = \sum_{i=1}^{m} l_i \, p_i i = \sum_{i=1}^{m-2} l'_i p_i + (l'_{m-1} + 1) p_{m-1} + (l'_{m-1} + 1) p_m$$

$$= \sum_{i=1}^{m-1} l_i p'_i + p_{m-1} + p_m = L_{m-1} + p_{m-1} + p_m$$

JPEG: Huffman Coding

- We observe that the average length of the code for m symbols is the average length of the code for m-1 symbols plus a constant. So, in order to minimize L_m we need to minimize L_{m-1} . We can use this fact recursively to get smaller and smaller alphabet.
- Consider for example a source with letters A, B, ..., G with probabilities {3/8, 3/16, 3/16, 1/8, 1/16, 1/32, 1/32}.

- L=2.44 bits/symbol.
- H(X)= 2.37 bits/symbol



THANK YOU FOR YOUR PATIENCE HEARING