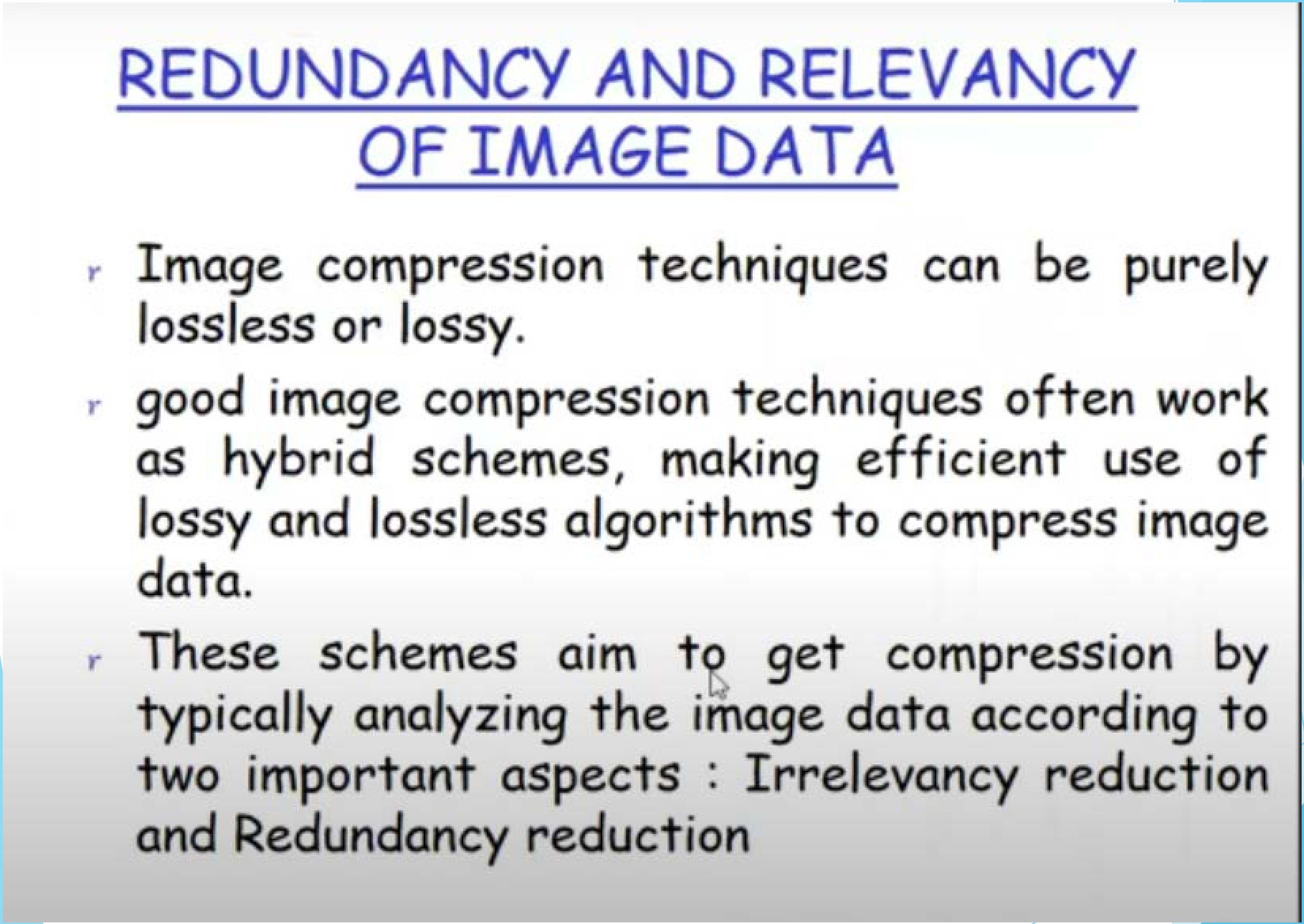
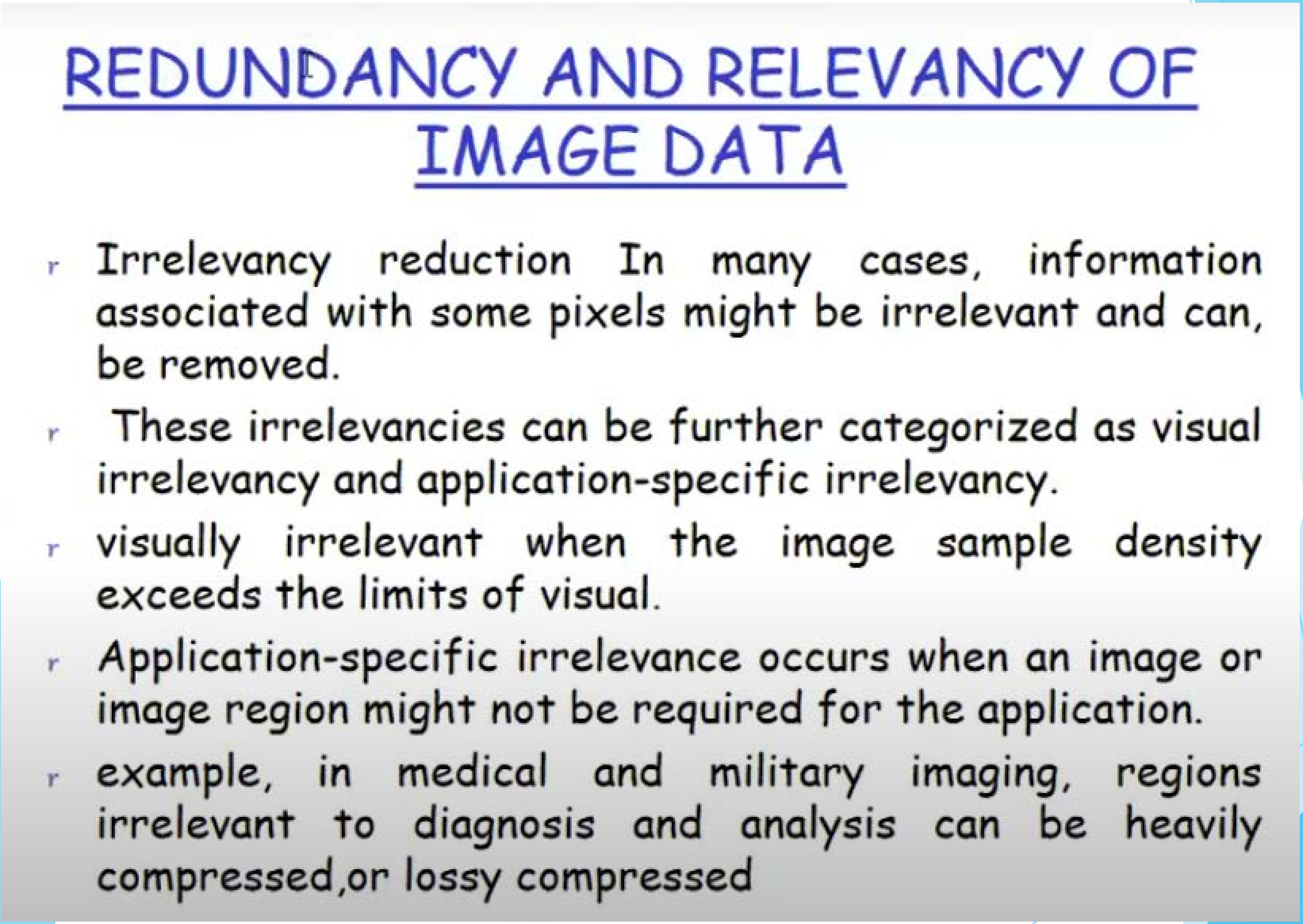
JPEG Compression



REDUNDANCY AND RELEVANCY OF IMAGE DATA

- Image compression techniques can be purely lossless or lossy.
- good image compression techniques often work as hybrid schemes, making efficient use of lossy and lossless algorithms to compress image data.
- These schemes aim to get compression by typically analyzing the image data according to two important aspects : Irrelevancy reduction and Redundancy reduction

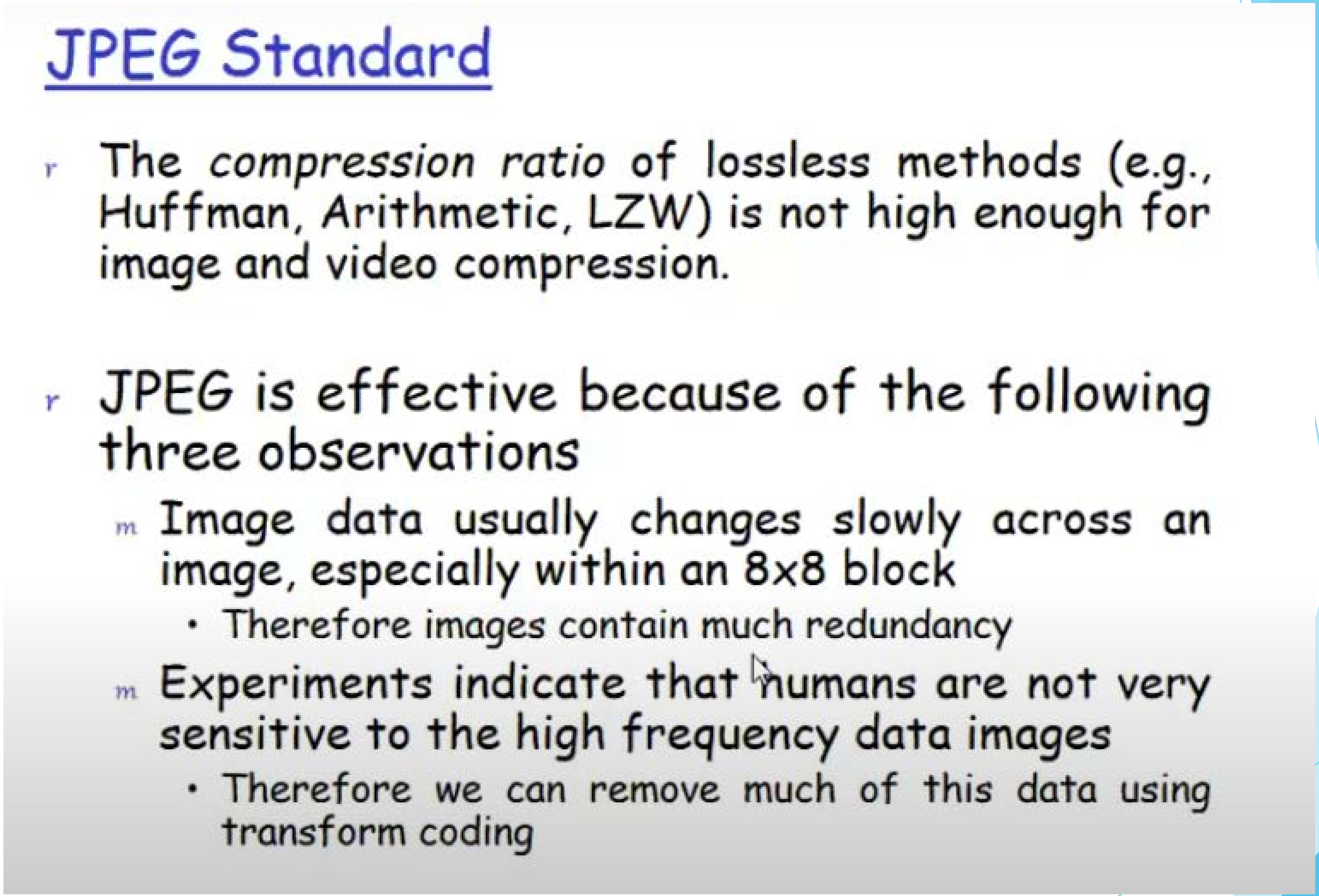


REDUNDANCY AND RELEVANCY OF IMAGE DATA

- Irrelevancy reduction In many cases, information associated with some pixels might be irrelevant and can, be removed.
- These irrelevancies can be further categorized as visual irrelevancy and application-specific irrelevancy.
- visually irrelevant when the image sample density exceeds the limits of visual.
- Application-specific irrelevance occurs when an image or image region might not be required for the application.
- example, in medical and military imaging, regions irrelevant to diagnosis and analysis can be heavily compressed,or lossy compressed

What is JPEG?

- **JPEG: Joint Photographic Expert Group** — an international standard in 1992.
- Works for both color and grayscale images.
- Targets at natural images.
- Applications include satellite, medical imaging, general photography ...



JPEG Standard

- The *compression ratio* of lossless methods (e.g., Huffman, Arithmetic, LZW) is not high enough for image and video compression.

- JPEG is effective because of the following three observations
  - Image data usually changes slowly across an image, especially within an 8x8 block
    - Therefore images contain much redundancy
  - Experiments indicate that humans are not very sensitive to the high frequency data images
    - Therefore we can remove much of this data using transform coding

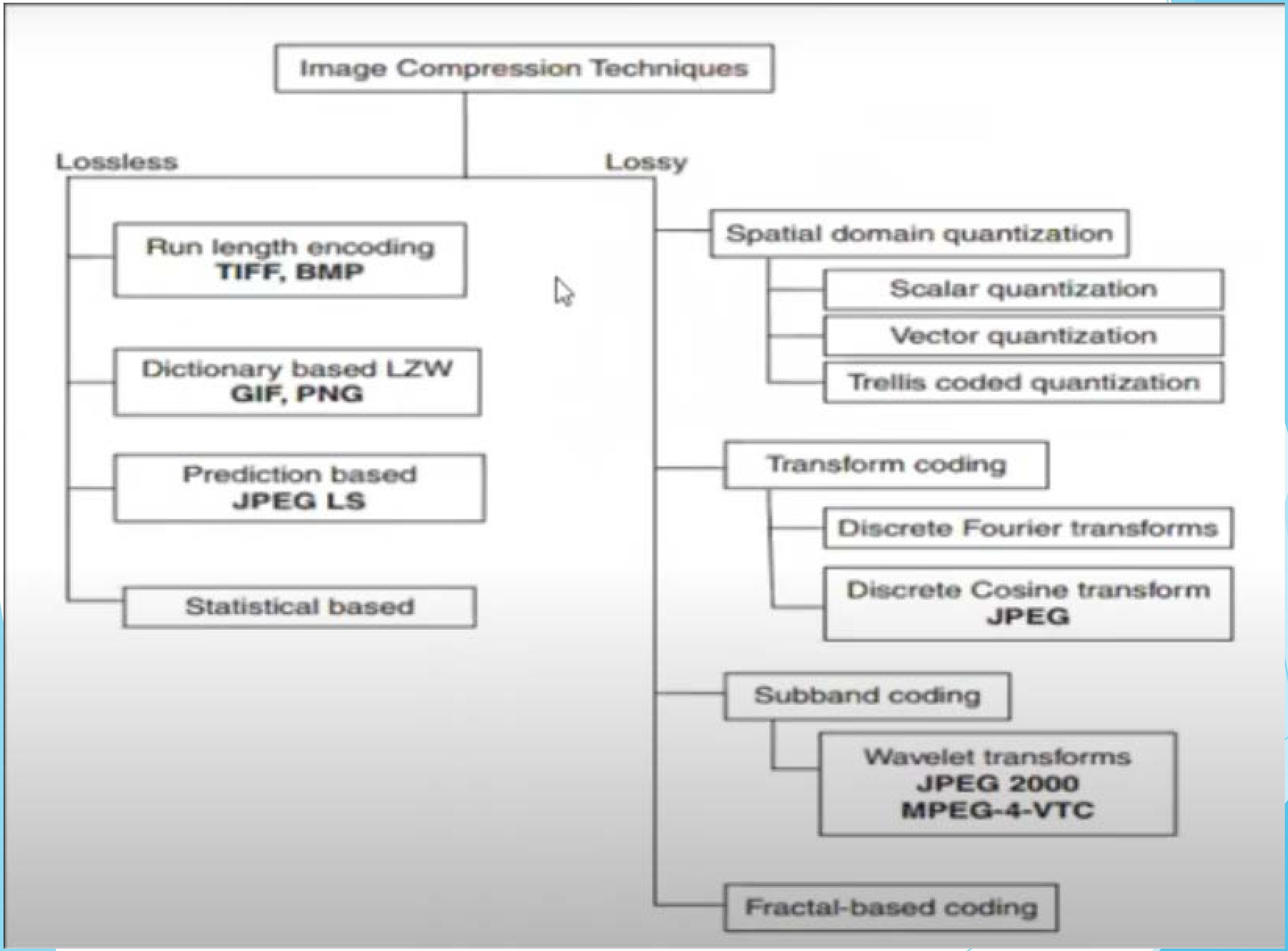
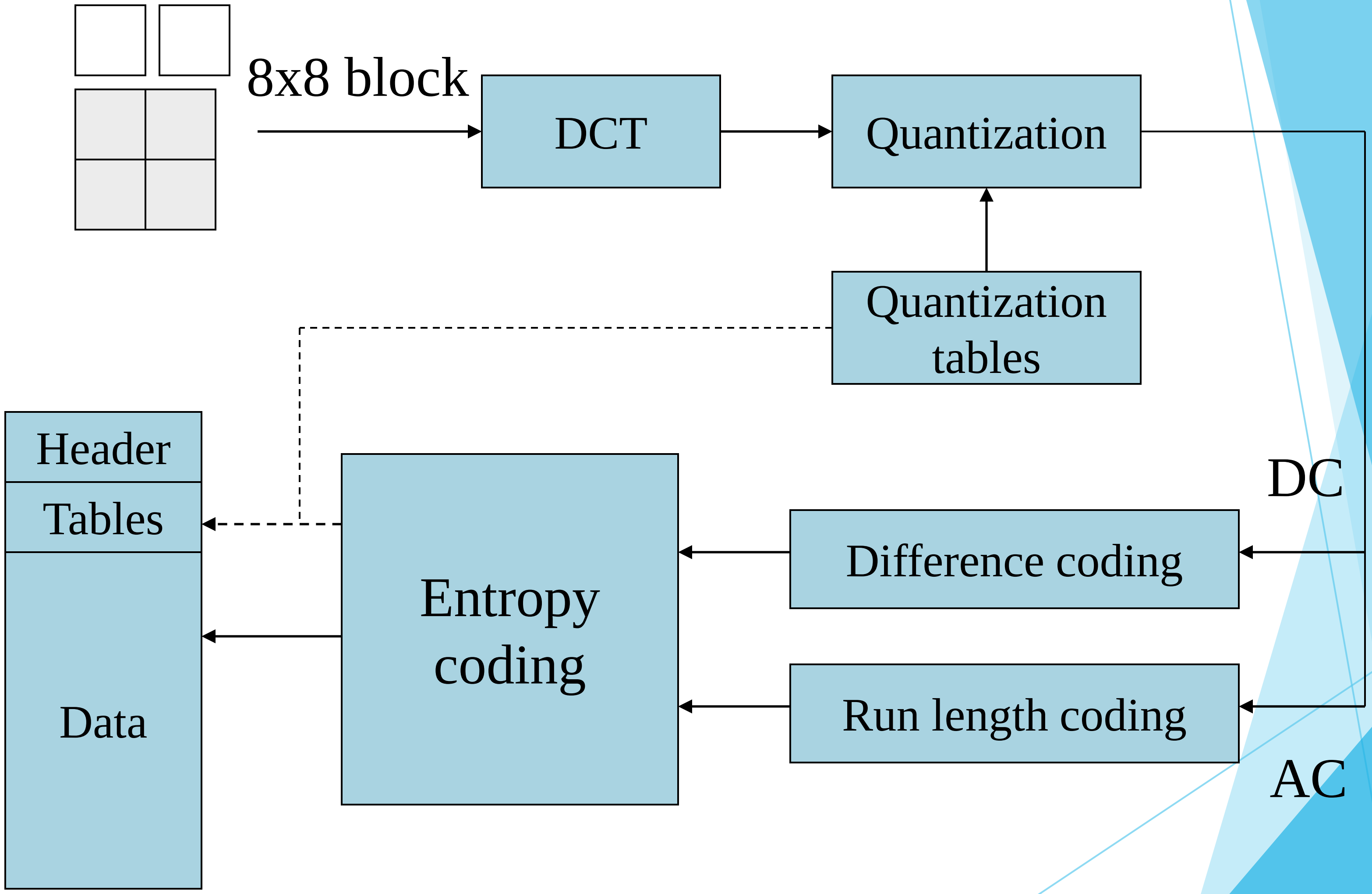
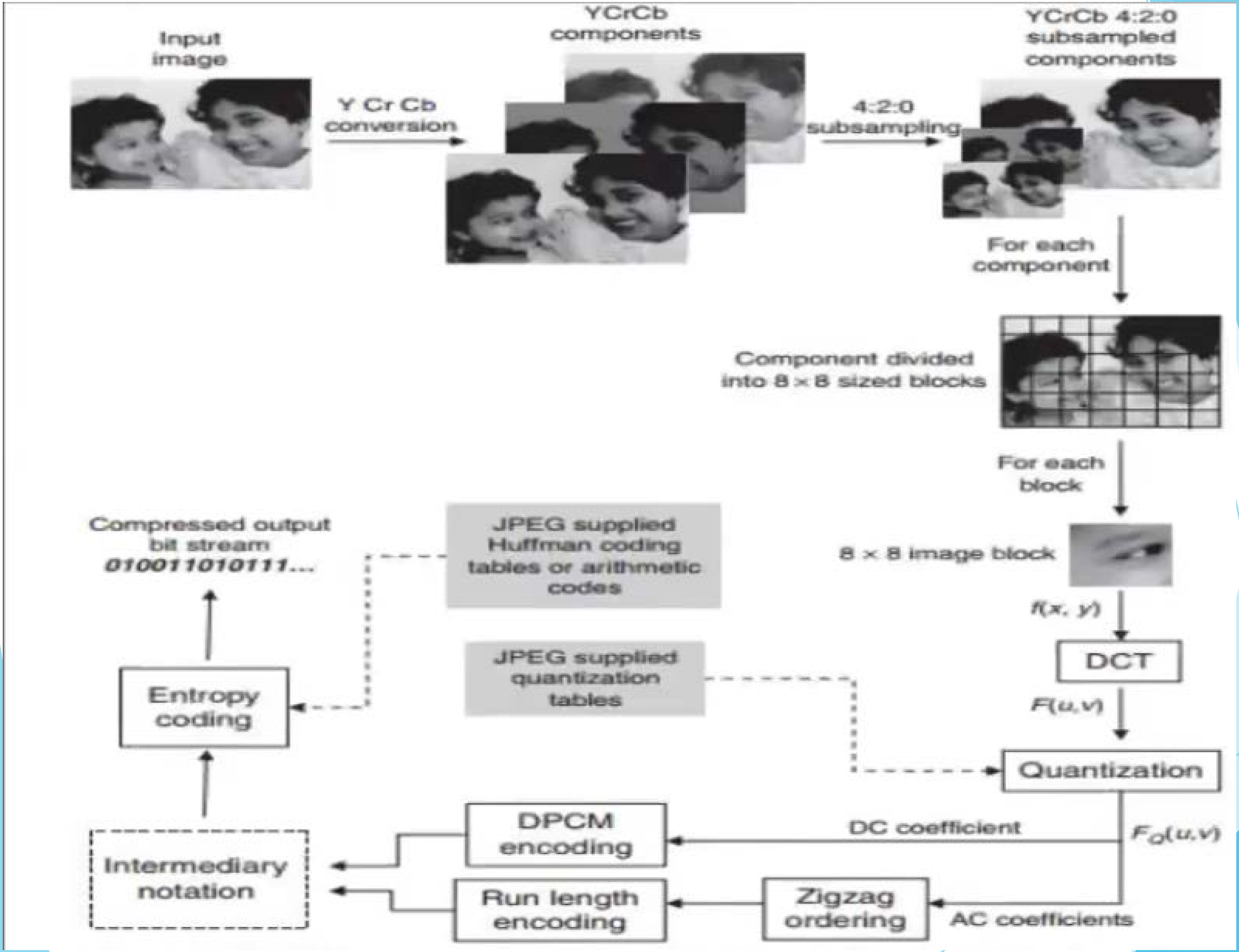


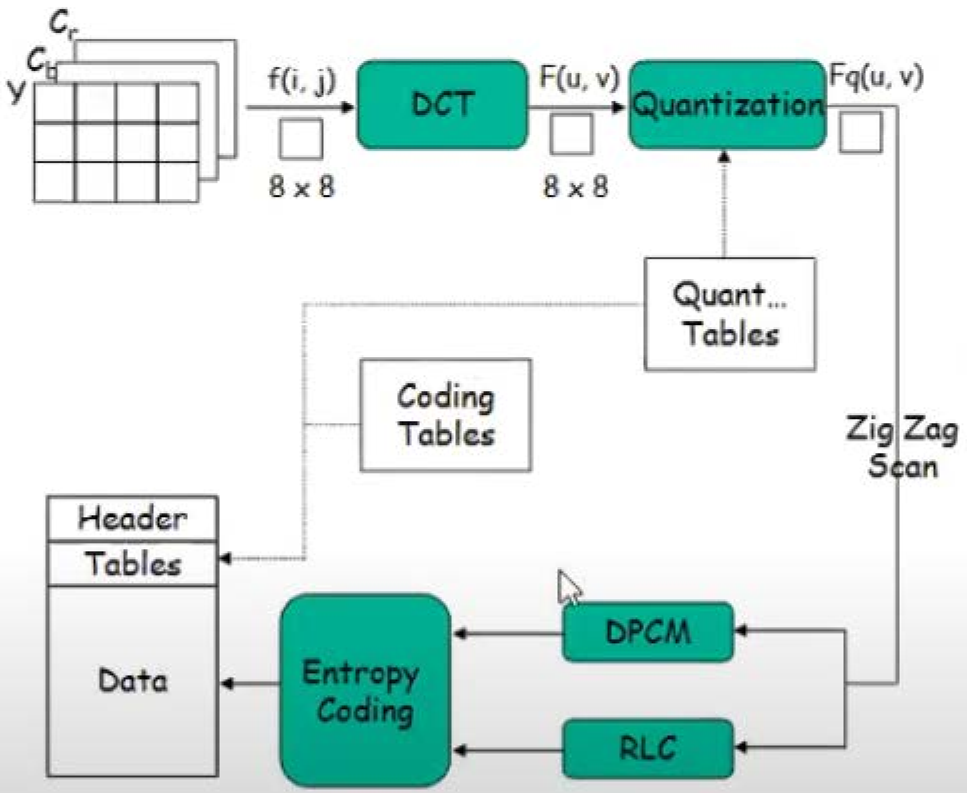
Image Compression Techniques
Lossless
Lossy
Run length encoding TIFF, BMP
Dictionary based LZW GIF, PNG
Prediction based JPEG LS
Statistical based
Spatial domain quantization
Scalar quantization
Vector quantization
Trellis coded quantization
Transform coding
Discrete Fourier transforms
Discrete Cosine transform JPEG
Subband coding
Wavelet transforms JPEG 2000 MPEG-4-VTC
Fractal-based coding

The System of JPEG




Input image
Y Cr Cb conversion
YCrCb components
4:2:0 subsampling
YCrCb 4:2:0 subsampled components
For each component
Component divided into 8 × 8 sized blocks
For each block
8 × 8 image block
f(x, y)
DCT
F(u,v)
Quantization
JPEG supplied quantization tables
F_Q(u,v)
DC coefficient
AC coefficients
Zigzag ordering
Run length encoding
DPCM encoding
Intermediary notation
Entropy coding
JPEG supplied Huffman coding tables or arithmetic codes
Compressed output bit stream
010011010111...

JPEG Coding



Steps Involved:

1. Discrete Cosine Transform of each 8x8 pixel array $f(x,y) \rightarrow_T F(u,v)$
2. Quantization using a table or using a constant
3. Zig-Zag scan to exploit redundancy
4. Differential Pulse Code Modulation(DPCM) on the DC component and Run length Coding of the AC components
5. Entropy coding (Huffman) of the final output

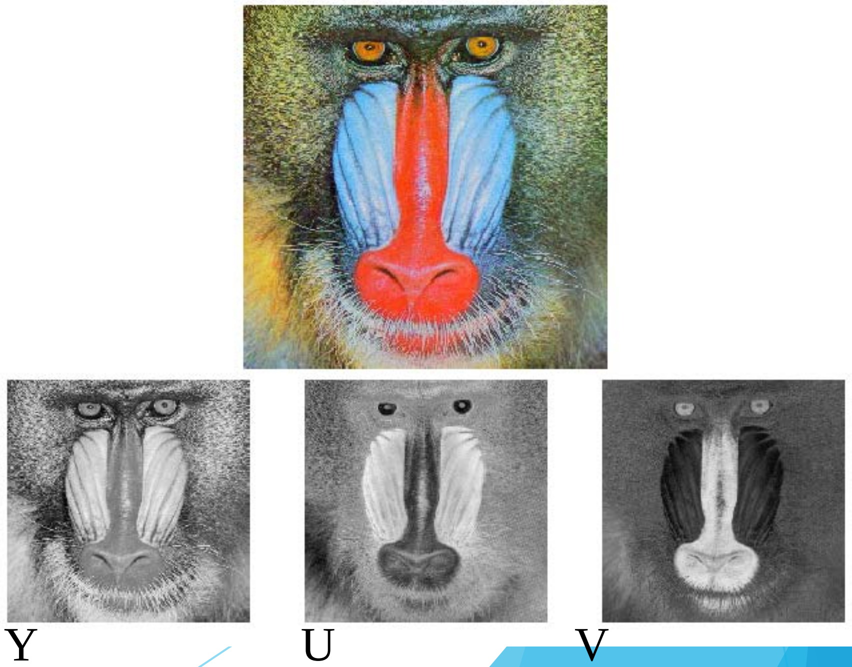
Color Space Conversion

- JPEG first converts RGB to YUV or YCrCb.
- Y is the luminance component (brightness).

  $Y = 0.299 R + 0.587 G + 0.144 B$
- U and V are color components

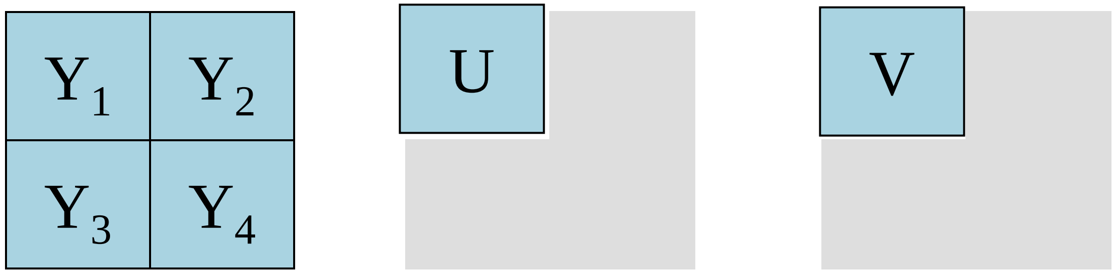
  $U = B – Y$

  $V = R - Y$



Color Subsampling

- JPEG down-samples the “color channels” by half and partitions images into 8x8 small blocks.



Intensity Component Color components

DCT

- DCT is the most popular transform method for image and video coding.
- It is efficient for transforming frame data into a form that is effortless to compress.
- Usually DCT is applied to a discrete blocks of an image. Each block is $2^n x 2^n$ pixels. The integer value of n is selected to determine the block size. For instance, if n=3, the block size will be 8x8 pixels.
- As moved to the bottom right corner of the blocks, the energy will be decreased.
- Hence, the transformed coefficients at the bottom right corner represent the higher frequency coefficients

DCT Coefficients

- The peak of the energy is concentrated in the transformed coefficients about the top-left corner of each block. The top-left corner coefficients match the DC and the low AC frequencies.

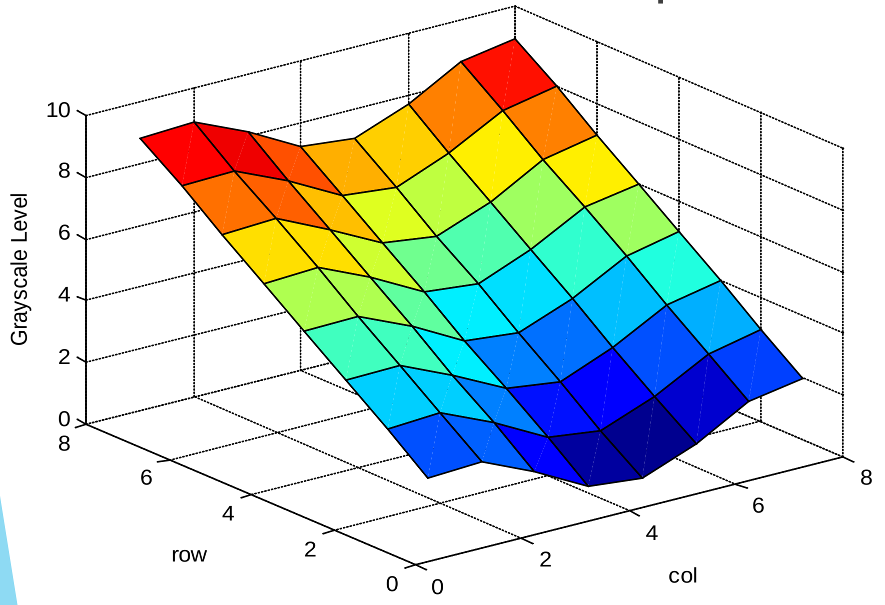
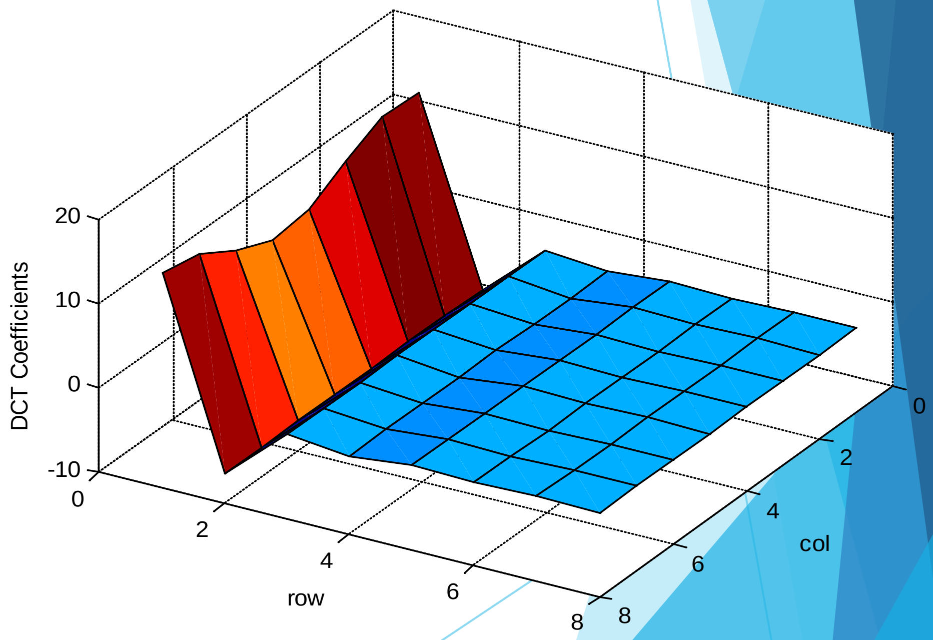


Image block

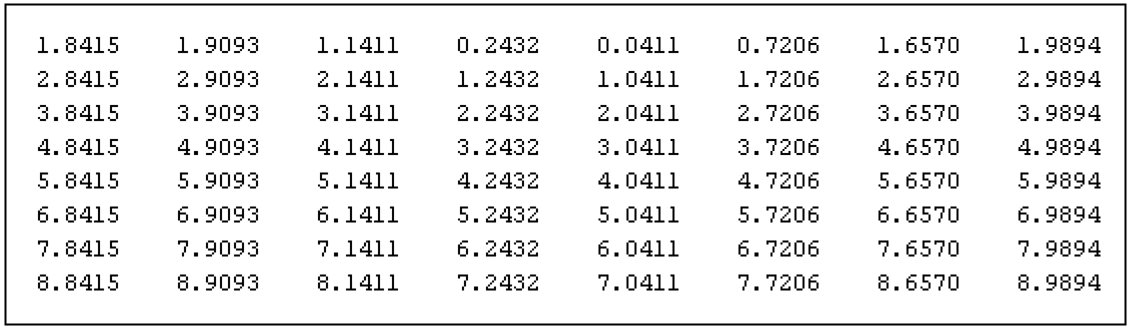


DCT Coefficients

DCT (Discrete Cosine Transform)

- DCT converts each image 8x8 block into another 8x8 block.
- The energy in DCT domain is concentrated into very few coefficients.

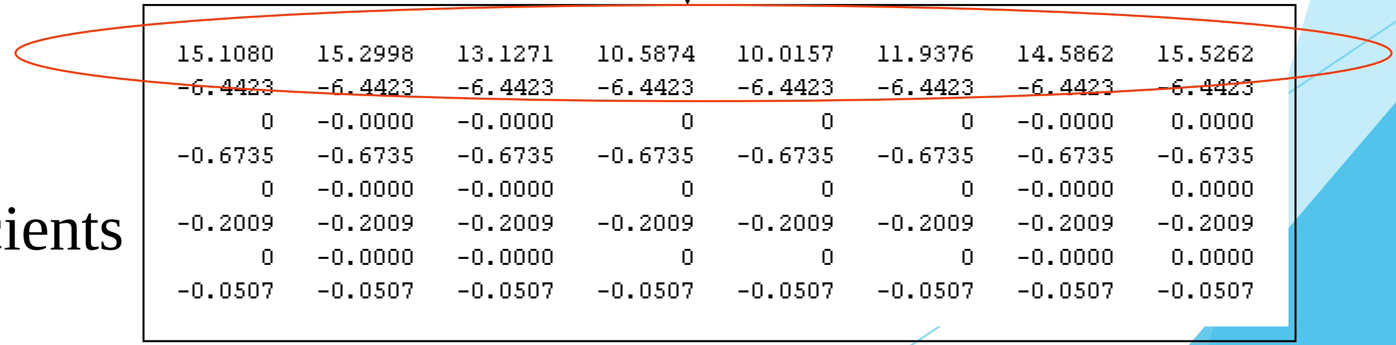
Image block



1.8415	1.9093	1.1411	0.2432	0.0411	0.7206	1.6570	1.9894
2.8415	2.9093	2.1411	1.2432	1.0411	1.7206	2.6570	2.9894
3.8415	3.9093	3.1411	2.2432	2.0411	2.7206	3.6570	3.9894
4.8415	4.9093	4.1411	3.2432	3.0411	3.7206	4.6570	4.9894
5.8415	5.9093	5.1411	4.2432	4.0411	4.7206	5.6570	5.9894
6.8415	6.9093	6.1411	5.2432	5.0411	5.7206	6.6570	6.9894
7.8415	7.9093	7.1411	6.2432	6.0411	6.7206	7.6570	7.9894
8.8415	8.9093	8.1411	7.2432	7.0411	7.7206	8.6570	8.9894

$\downarrow$ DCT

DCT coefficients



15.1080	15.2998	13.1271	10.5874	10.0157	11.9376	14.5862	15.5262
-6.4423	-6.4423	-6.4423	-6.4423	-6.4423	-6.4423	-6.4423	-6.4423
0	-0.0000	-0.0000	0	0	0	-0.0000	0.0000
-0.6735	-0.6735	-0.6735	-0.6735	-0.6735	-0.6735	-0.6735	-0.6735
0	-0.0000	-0.0000	0	0	0	-0.0000	0.0000
-0.2009	-0.2009	-0.2009	-0.2009	-0.2009	-0.2009	-0.2009	-0.2009
0	-0.0000	-0.0000	0	0	0	-0.0000	0.0000
-0.0507	-0.0507	-0.0507	-0.0507	-0.0507	-0.0507	-0.0507	-0.0507

Definition of DCT

A linear transfrom $F(u,v) = \sum_{x=0}^{7} \sum_{y=0}^{7} h(u,v,x,y)\, f(x,y)$

where $h(u,v,x,y)$ is the linear weighting function.

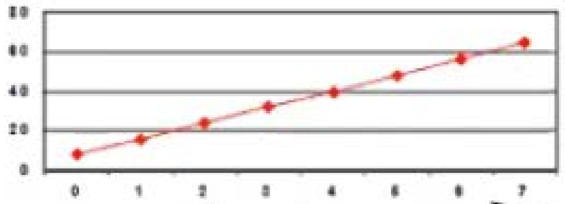
DCT:

$$F(u,v) = \frac{2C(u)C(v)}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \cos\frac{(2x+1)u\pi}{2M} \cos\frac{(2y+1)v\pi}{2N} f(x,y)$$

$$C(\theta) = \begin{cases} \sqrt{2}/2 & \theta = 0 \\ 1 & otherwise \end{cases}$$

The inverse transform IDCT recovers the original data from DCT coefficients.

Example and Comparison



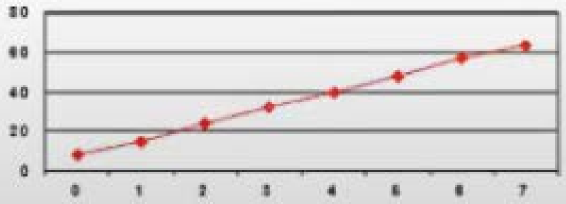
DCT

100	-52	0	-5	0	-2	0	0.4

100	-52	0	-5	~~0~~	~~-2~~	~~0~~	~~0.4~~

Inverse DCT

8	15	24	32	40	48	57	63



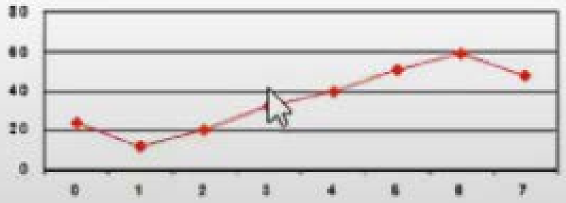
FFT

36	10	10	6	6	4	4	4

36	10	10	6	~~6~~	~~4~~	~~4~~	~~4~~

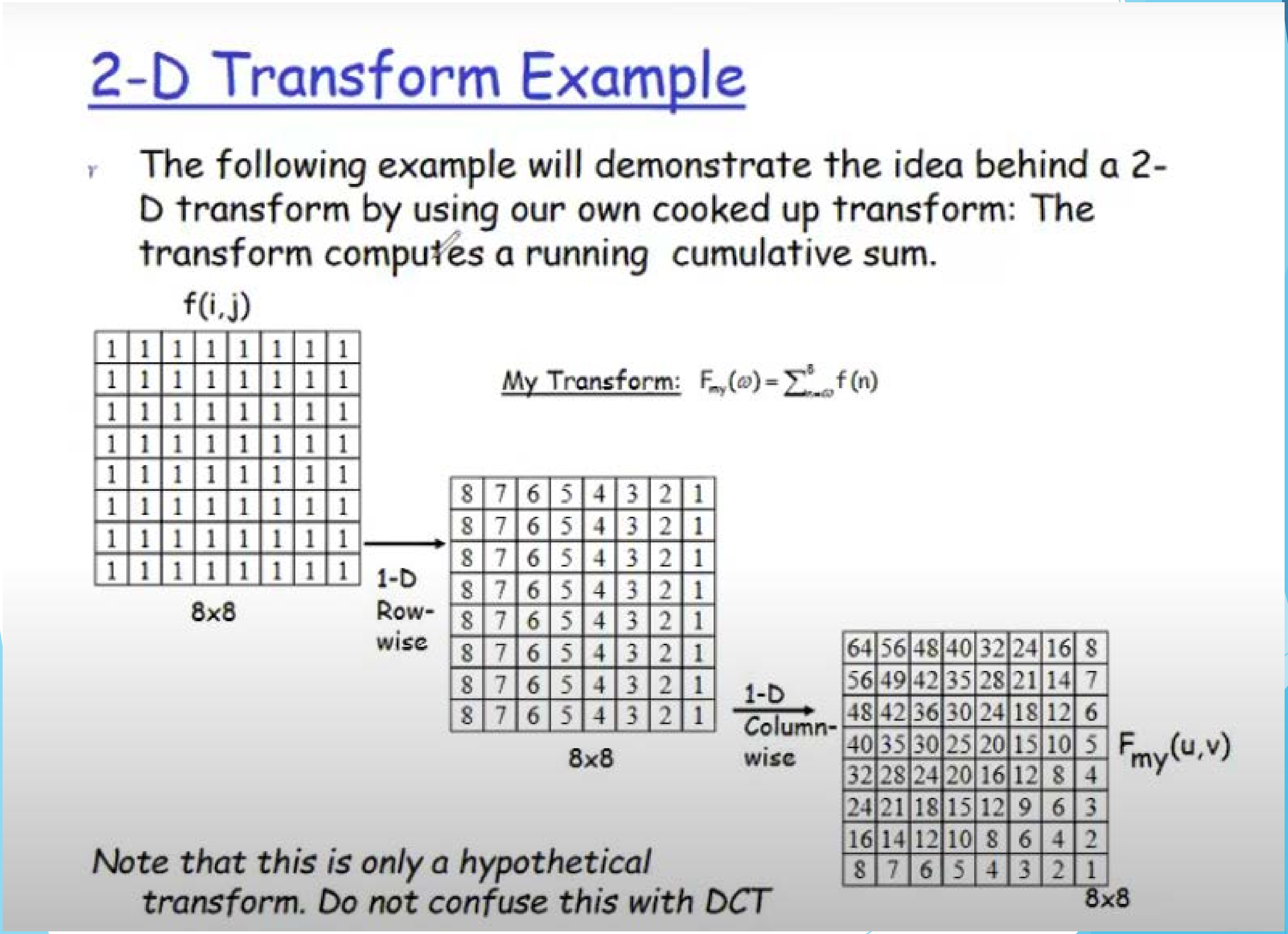
Inverse FFT

24	12	20	32	40	51	59	48



Example Description:

- f(n) is given from n = 0 to 7; (N=8)
- Using DCT(FFT) we compute F(ω) for ω = 0 to 7
- We truncate and use Inverse Transform to compute f'(n)



2-D Transform Example

- The following example will demonstrate the idea behind a 2-D transform by using our own cooked up transform: The transform computes a running cumulative sum.

f(i,j)

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

8x8

<u>My Transform:</u> $F_{my}(\omega) = \sum_{n=\omega}^{8} f(n)$

$\xrightarrow{\text{1-D Row-wise}}$

8	7	6	5	4	3	2	1
8	7	6	5	4	3	2	1
8	7	6	5	4	3	2	1
8	7	6	5	4	3	2	1
8	7	6	5	4	3	2	1
8	7	6	5	4	3	2	1
8	7	6	5	4	3	2	1
8	7	6	5	4	3	2	1

8x8

$\xrightarrow{\text{1-D Column-wise}}$

64	56	48	40	32	24	16	8
56	49	42	35	28	21	14	7
48	42	36	30	24	18	12	6
40	35	30	25	20	15	10	5
32	28	24	20	16	12	8	4
24	21	18	15	12	9	6	3
16	14	12	10	8	6	4	2
8	7	6	5	4	3	2	1

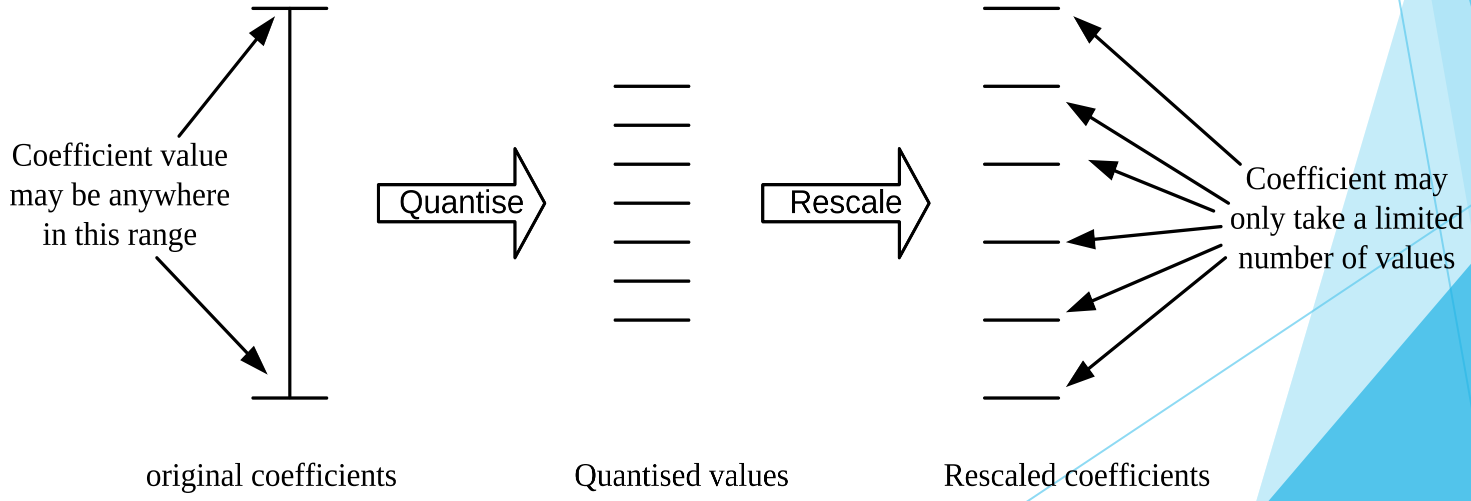
$F_{my}(u,v)$

8x8

Note that this is only a hypothetical transform. Do not confuse this with DCT

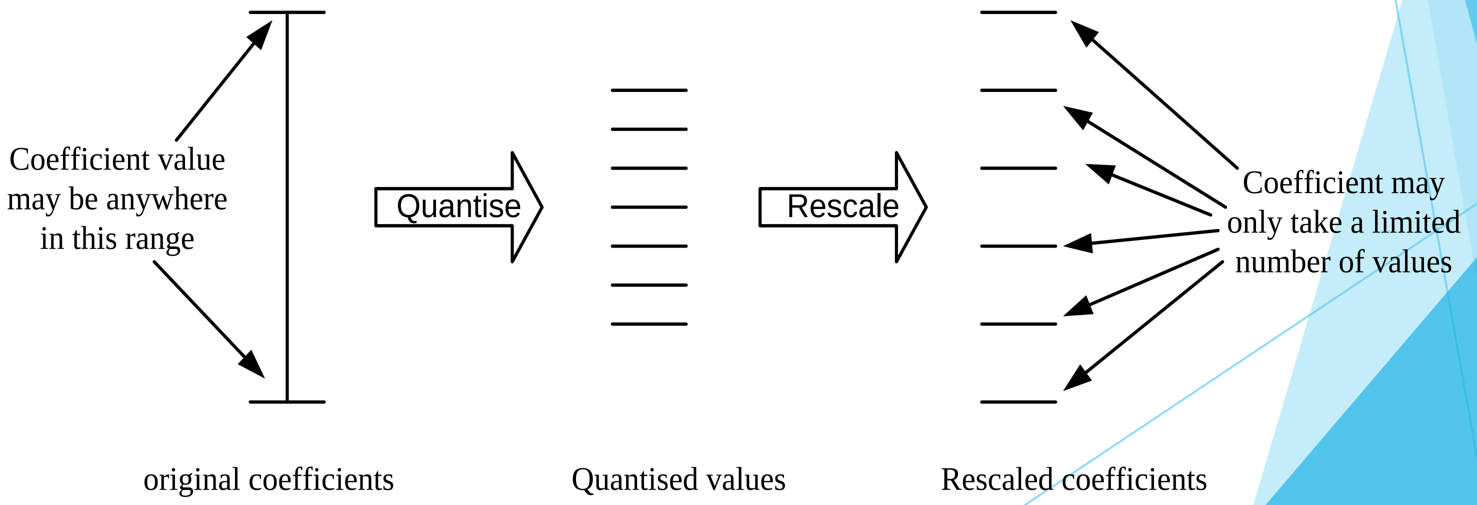
Quantization Process

- Quantization follows the transform stage.
- It aims to removes the coefficients that do not have a significant influence on the appearance of the reconstructed image.
- Thus, it represents the actual first step in the compression process. The quantization process is split into two operations, as shown in figure.


Coefficient value may be anywhere in this range
Quantise
Rescale
Coefficient may only take a limited number of values
original coefficients
Quantised values
Rescaled coefficients

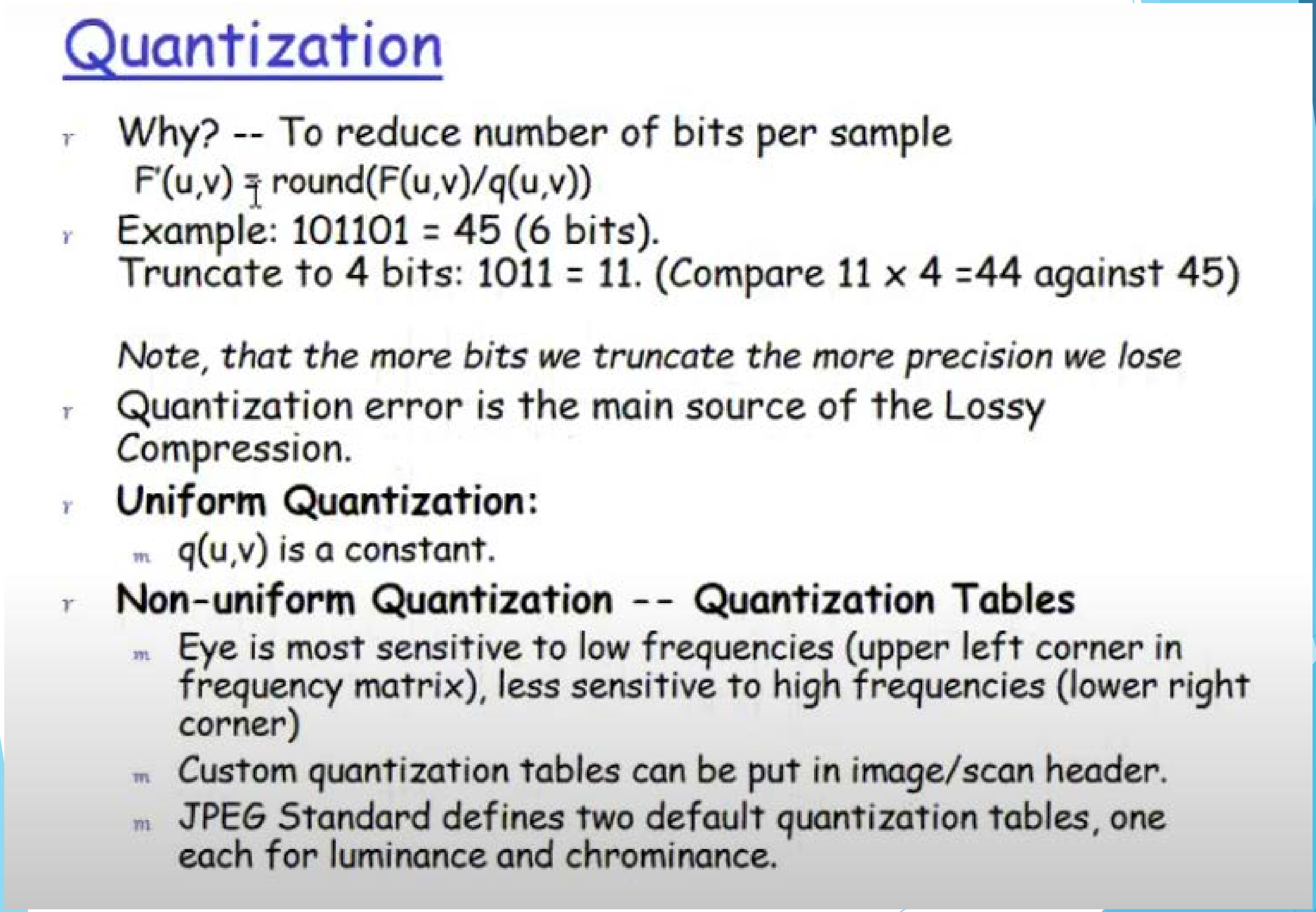
Quantization Process

- The first quantization operation is performed during the encoding. It converts the transformed coefficients into a limited number of possible levels (quantization).
- The second operation is performed during the decoding. The levels are rescaled, to produce the reconstructed coefficients (rescaling or inverse quantization). These coefficients are approximately having the same magnitude as the original coefficients.


Coefficient value may be anywhere in this range
Quantise
Rescale
Coefficient may only take a limited number of values
original coefficients
Quantised values
Rescaled coefficients

Quantization Results

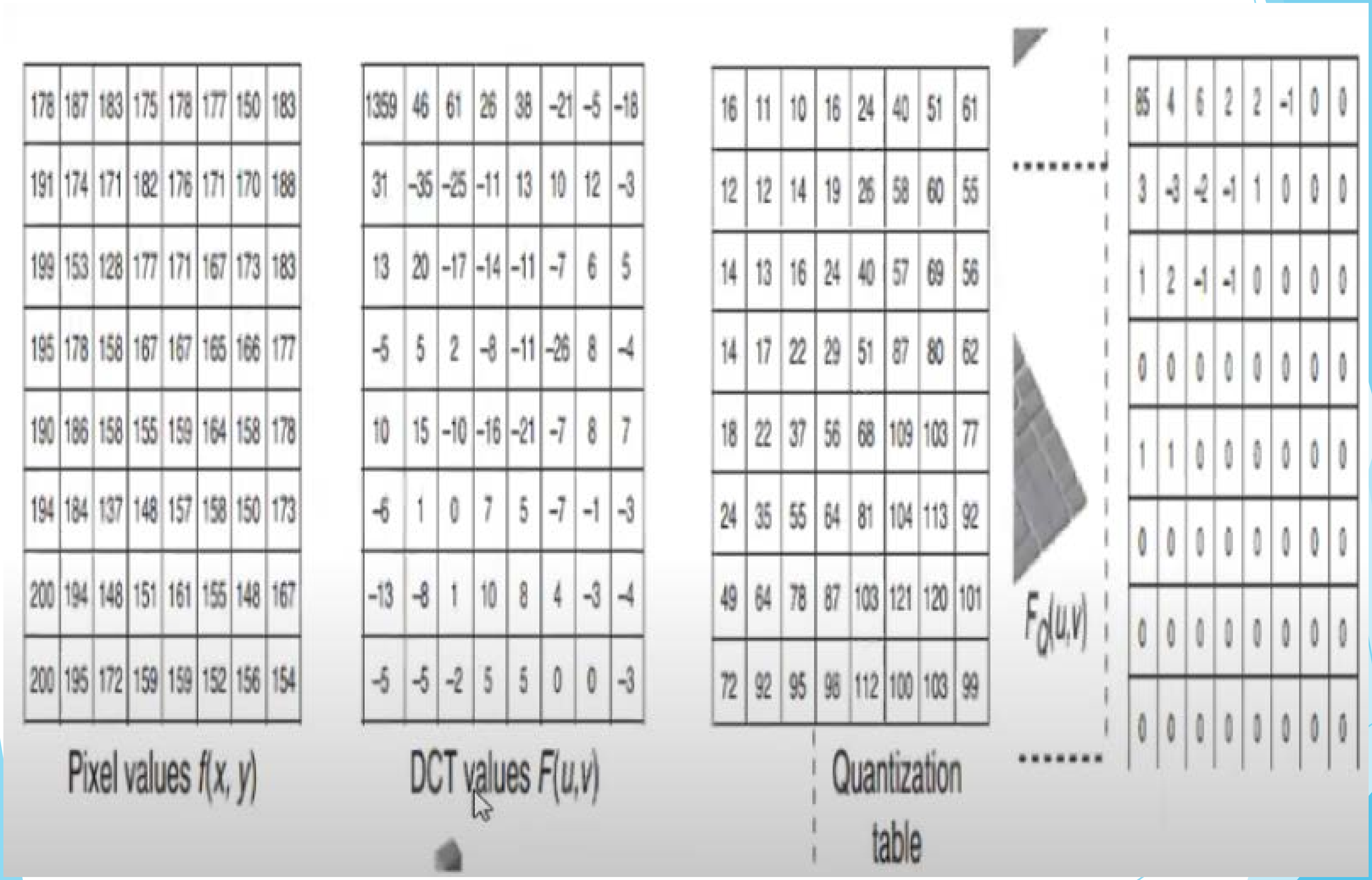
- The visual significant coefficients are retained, while the insignificant coefficients are discarded.
- A sparse matrix containing levels with a limited number of discrete values (the result of the quantization) can be efficiently compressed.
- The reconstructed coefficients are not identical to the original. Thus, there is a harmful effect to the image quality due to quantization.
- The amount of compression and the loss of image quality depend on the number of levels produced by the quantization.
- A large number of the levels mean the coefficient precision is only slightly reduced and the compression is low



Quantization

- Why? -- To reduce number of bits per sample
  $F'(u,v) = \text{round}(F(u,v)/q(u,v))$
- Example: 101101 = 45 (6 bits).
  Truncate to 4 bits: 1011 = 11. (Compare 11 x 4 =44 against 45)

  Note, that the more bits we truncate the more precision we lose
- Quantization error is the main source of the Lossy Compression.
- **Uniform Quantization:**
  - q(u,v) is a constant.
- **Non-uniform Quantization -- Quantization Tables**
  - Eye is most sensitive to low frequencies (upper left corner in frequency matrix), less sensitive to high frequencies (lower right corner)
  - Custom quantization tables can be put in image/scan header.
  - JPEG Standard defines two default quantization tables, one each for luminance and chrominance.



178	187	183	175	178	177	150	183
191	174	171	182	176	171	170	188
199	153	128	177	171	167	173	183
195	178	158	167	167	165	166	177
190	186	158	155	159	164	158	178
194	184	137	148	157	158	150	173
200	194	148	151	161	155	148	167
200	195	172	159	159	152	156	154

Pixel values $f(x, y)$

1359	46	61	26	38	−21	−5	−18
31	−35	−25	−11	13	10	12	−3
13	20	−17	−14	−11	−7	6	5
−5	5	2	−8	−11	−26	8	−4
10	15	−10	−16	−21	−7	8	7
−6	1	0	7	5	−7	−1	−3
−13	−8	1	10	8	4	−3	−4
−5	−5	−2	5	5	0	0	−3

DCT values $F(u,v)$

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

Quantization table

$F_Q(u,v)$

85	4	6	2	2	-1	0	0
3	-3	-2	-1	1	0	0	0
1	2	-1	-1	0	0	0	0
0	0	0	0	0	0	0	0
1	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

The Quality Factor

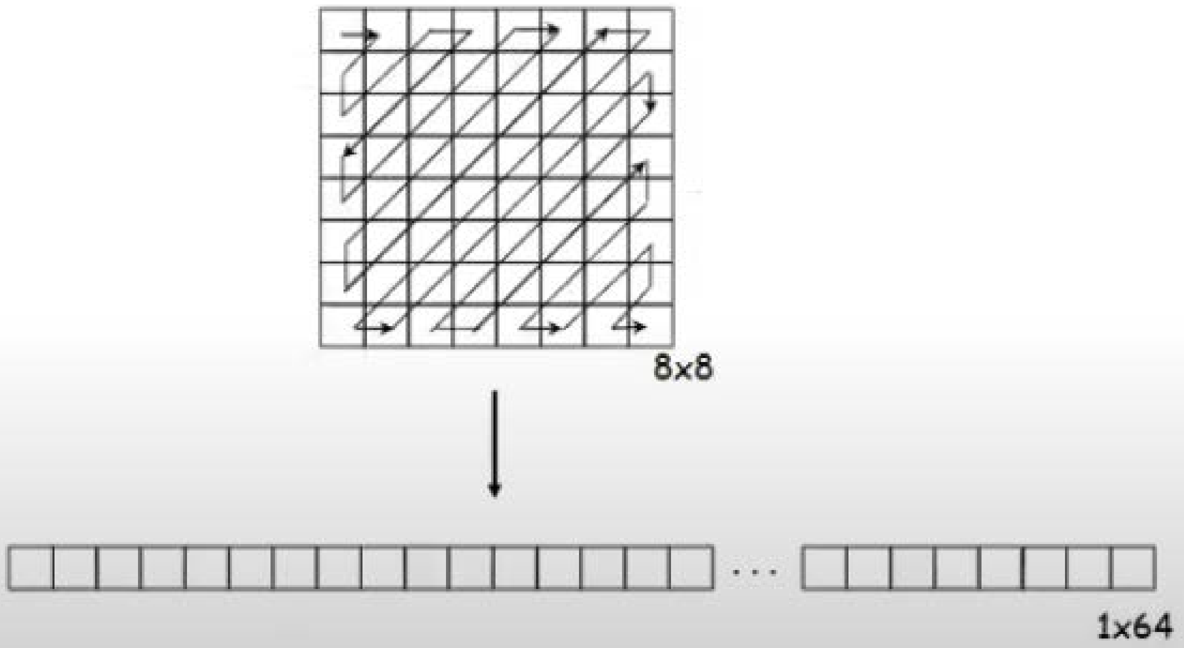
- The method of controlling the compression quality is by scaling the quantization table. For example,

$$F_q(u,v) = \mathrm{round}(F(u,v)^* \ (\mathrm{quality}) \ / \ Q(u,v))$$

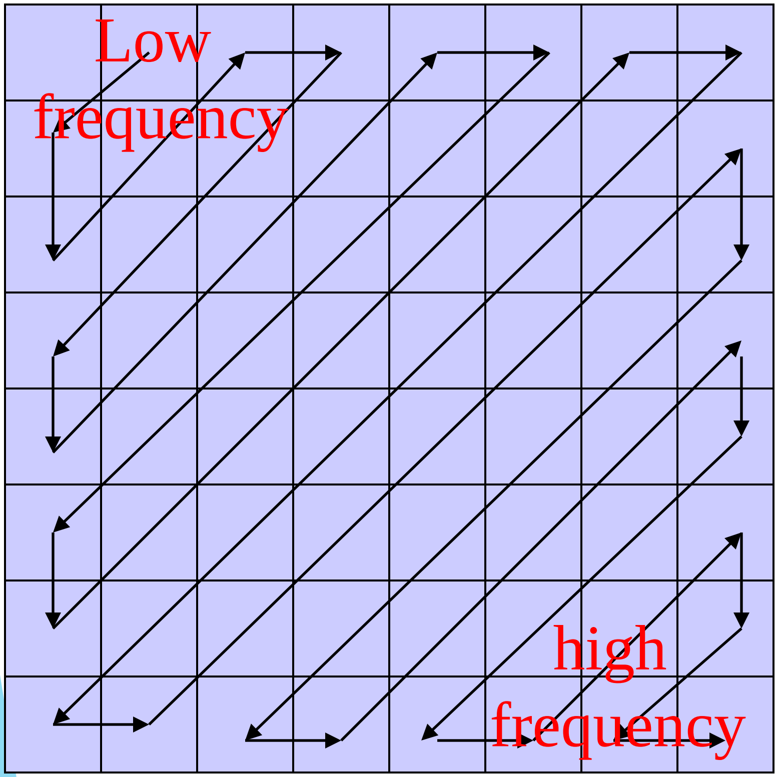
De-quantization: $F_r(u,v) = F_q(u,v) * Q(u,v)$

Zig-Zag Scan

- Why? -- to group low frequency coefficients in top of vector and high frequency coefficients at the bottom
- Maps 8 x 8 matrix to a 1 x 64 vector



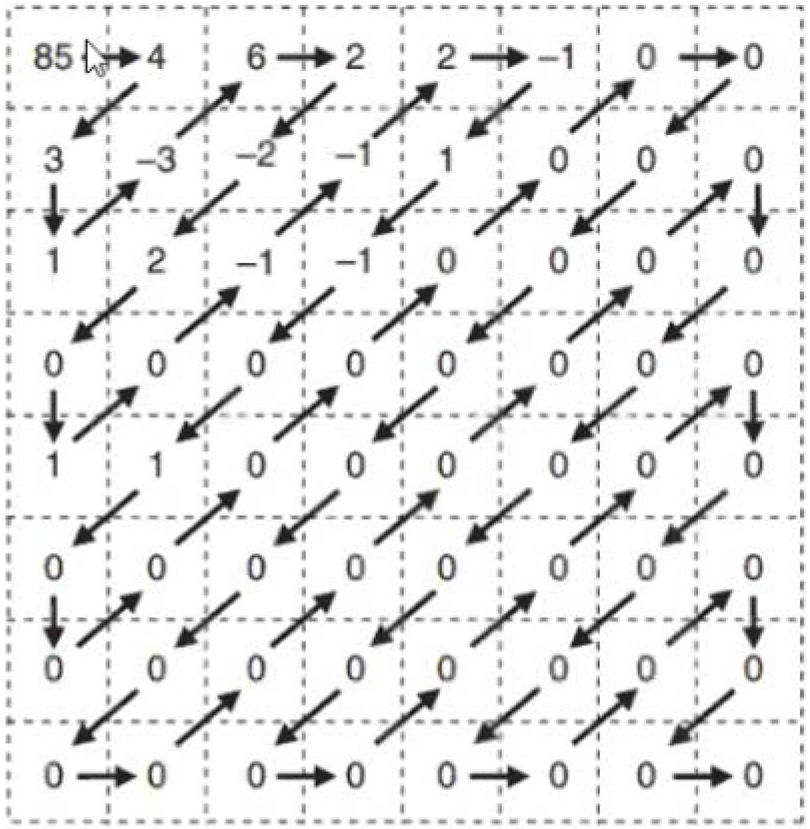
Zig-zag Reordering



Zig-zag scanning

- The output of the quantizer stage in a DCT-based encoder is a block of quantized transform coefficients.
- Most of the quantized coefficients in a block are zeros and the array of the coefficients is likely to be sparse.
- The 8x8 block of the quantized coefficients is re-arranged in zigzag order so that the low frequencies are grouped together at the start of the rearranged array, to be prepared for entropy coding as shown in figure

Zig-Zag Scan



85	4	6	2	2	−1	0	0
3	−3	−2	−1	1	0	0	0
1	2	−1	−1	0	0	0	0
0	0	0	0	0	0	0	0
1	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

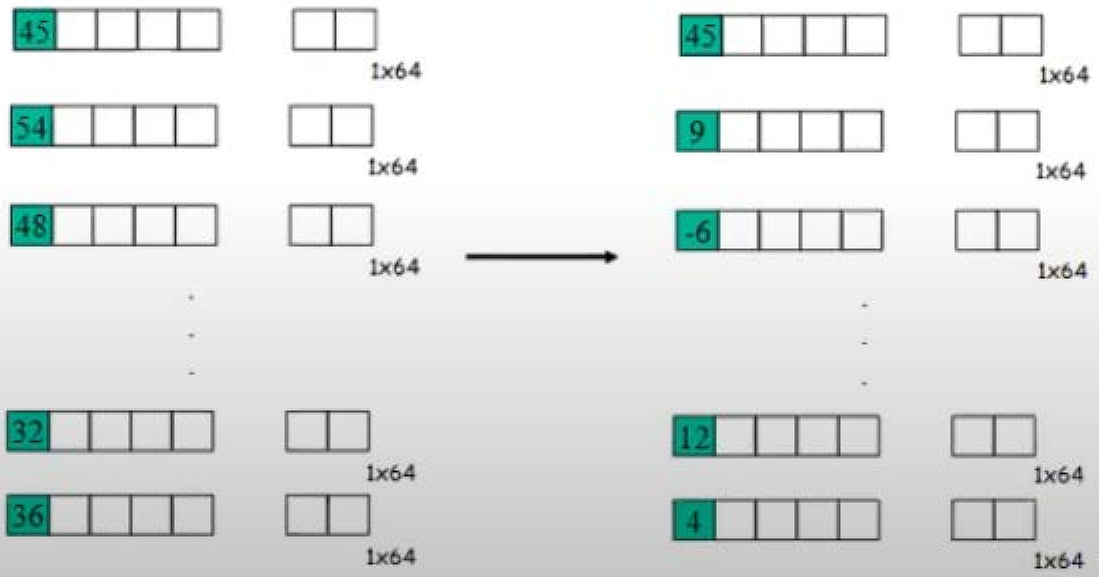
DC coefficient = 85

AC coefficient stream

4 3 1 -3 6 2 -2 2 0 1 0 -1 -1 2
-1 1 -1 0 1 0 0 0 0 0 0 0 0 0 ...

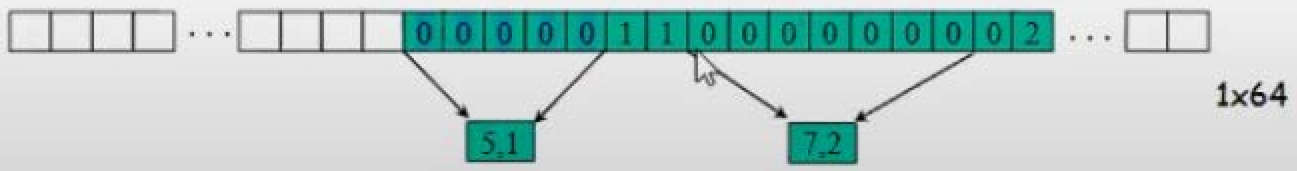
DPCM on DC Components

- The DC component value in each 8x8 block is large and varies across blocks, but is often close to that in the previous block.
- Differential Pulse Code Modulation (DPCM): Encode the difference between the current and previous 8x8 block. Remember, smaller number -> fewer bits



RLE on AC Components

- The 1x64 vectors have a lot of zeros in them, more so towards the end of the vector.
- Encode a series of 0s as a (*skip*,*value*) pair, where *skip* is the number of zeros and *value* is the next non-zero component.
  - Send (0,0) as end-of-block sentinel value.



Entropy Coding

- Entropy encoder is the second phase in the compression process.
- It maps input symbols to a compressed data stream .
- The reordered coefficient array is usually sparse, consisting of group of non-zero coefficients followed by zeros
- In order to realize the operation of an entropy encoder, we will deal with the non-zero coefficients shown in Table

102	**-33**	**-3**	**-4**	**-2**	**-1**	0	0
-21	**-2**	**-3**	0	**-1**	0	0	0
-3	0	**1**	0	0	0	0	0
2	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0
-2	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Table of quantized coefficients

Entropy Coding

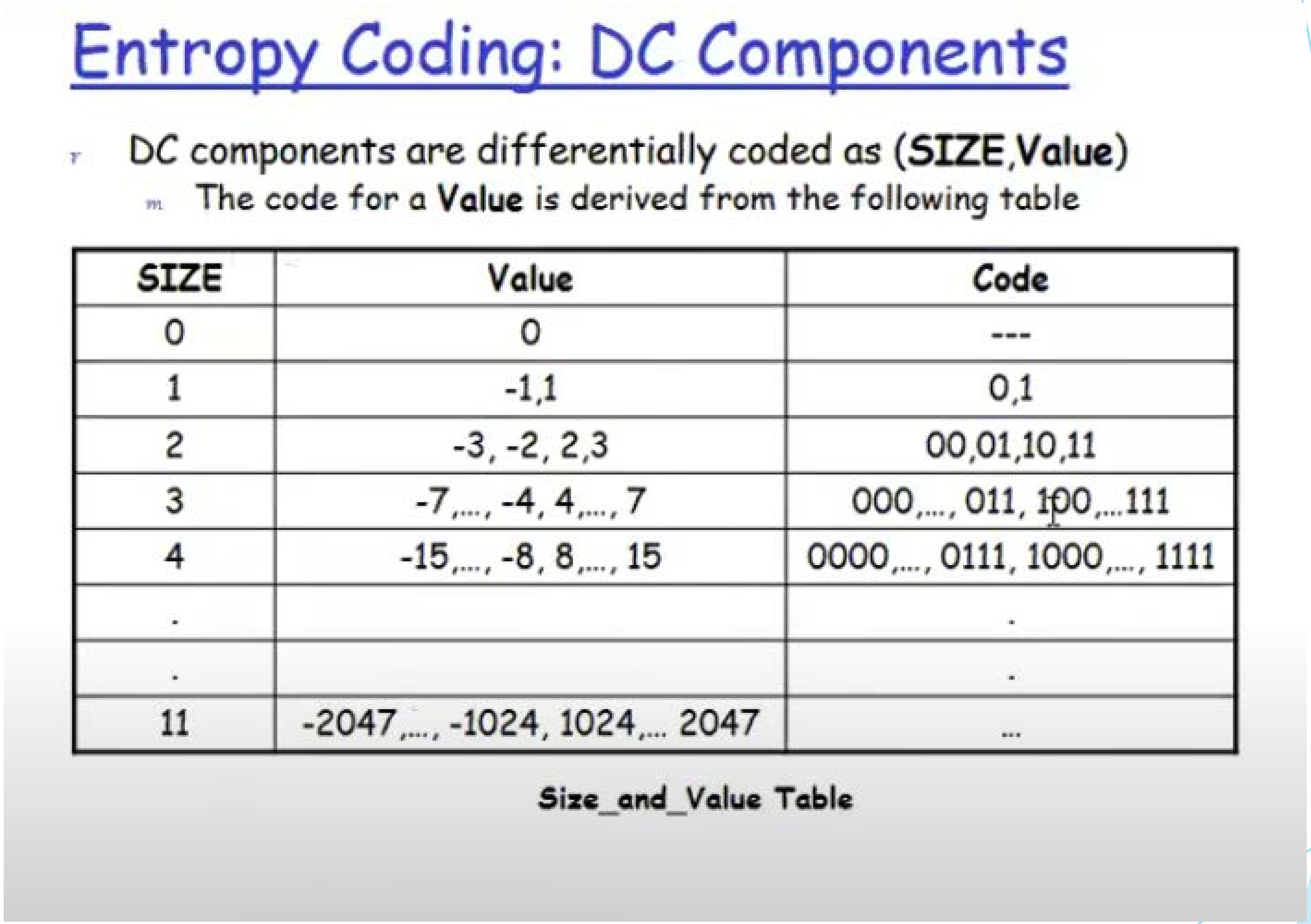
- The non-zero coefficients can be efficiently compressed using a statistical method as follows:
- **Re-order the quantized coefficients**; the non-zero quantized coefficients of each block are re-ordered in a zigzag scanning order.
- **Run-level coding**; for the group of non-zero coefficients that followed by zeros coefficients . a series of (run, level) pairs, as shown in Table are used to represent these coefficients. The first number of (run, level) pair represents the number of preceding zeros and the second number represents a non-zero value (level).

Reordered coefficient data	Run-level coded
102	(0,102)
-33	(0,-33)
-21	(0,-21)
-3	(0,-3)
-2	(0,-2)
-3	(0,-3)
-4	(0,-4)
-3	(0,-3)
0	(1,2)
….	….

Entropy coding; **a statistical algorithm is applied to the (run, level) data, to represent frequently occurring (run, level) pairs with a short code and infrequently occurring (run, level) pairs with a longer code. In this way, the run-level data may be compressed into a small number of bits.**

Decoding Process

- The decoding operations are the equivalent to inverse of the encoding operations.
- In order to recreate the image, an entropy decoder extracts run-level symbols from the received bit stream.
- These symbols are converted to a sequence of coefficients that are reordered into a block of quantized coefficients.
- Each coefficient is multiplied by an integer scale factor (rescaled).
- The loss of precision that occurred due to the quantization cannot be reversed. Hence, the rescaled coefficients are not identical to the original transform coefficients.
- The rescaled coefficients are inverse transformed to reconstruct a decoded image. As a result of the data loss during quantization, the reconstructed image will not be identical to the original image

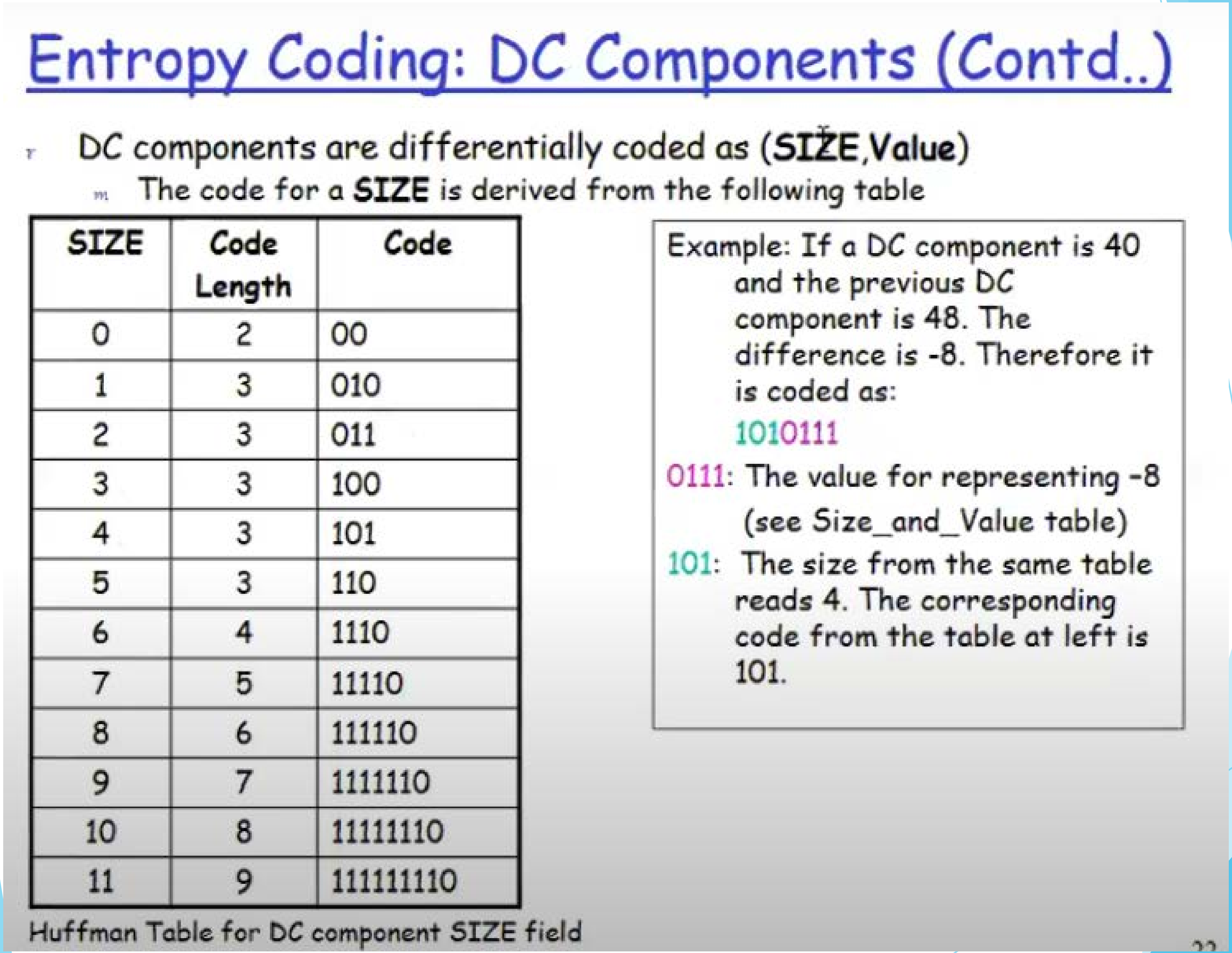


Entropy Coding: DC Components

- DC components are differentially coded as (**SIZE**,**Value**)
  - The code for a **Value** is derived from the following table

SIZE	Value	Code
0	0	---
1	-1,1	0,1
2	-3, -2, 2,3	00,01,10,11
3	-7,...., -4, 4,...., 7	000,...., 011, 100,...111
4	-15,...., -8, 8,...., 15	0000,...., 0111, 1000,...., 1111
.		.
.		.
11	-2047,...., -1024, 1024,... 2047	...

Size_and_Value Table



Entropy Coding: DC Components (Contd..)

- DC components are differentially coded as (**SIZE**,**Value**)
  - The code for a **SIZE** is derived from the following table

SIZE	Code Length	Code
0	2	00
1	3	010
2	3	011
3	3	100
4	3	101
5	3	110
6	4	1110
7	5	11110
8	6	111110
9	7	1111110
10	8	11111110
11	9	111111110

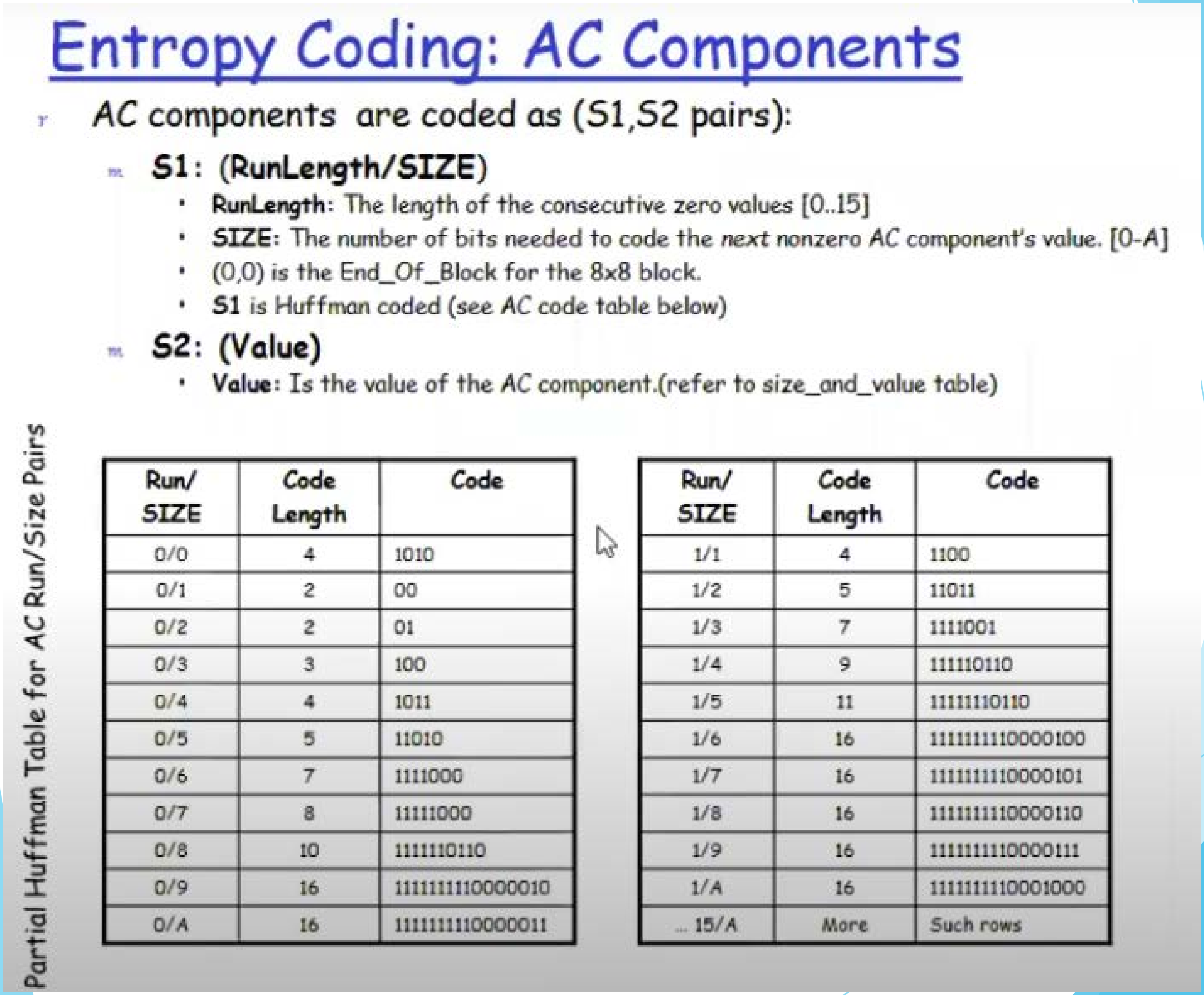
Huffman Table for DC component SIZE field

Example: If a DC component is 40 and the previous DC component is 48. The difference is -8. Therefore it is coded as:

1010111

0111: The value for representing -8 (see Size_and_Value table)

101: The size from the same table reads 4. The corresponding code from the table at left is 101.



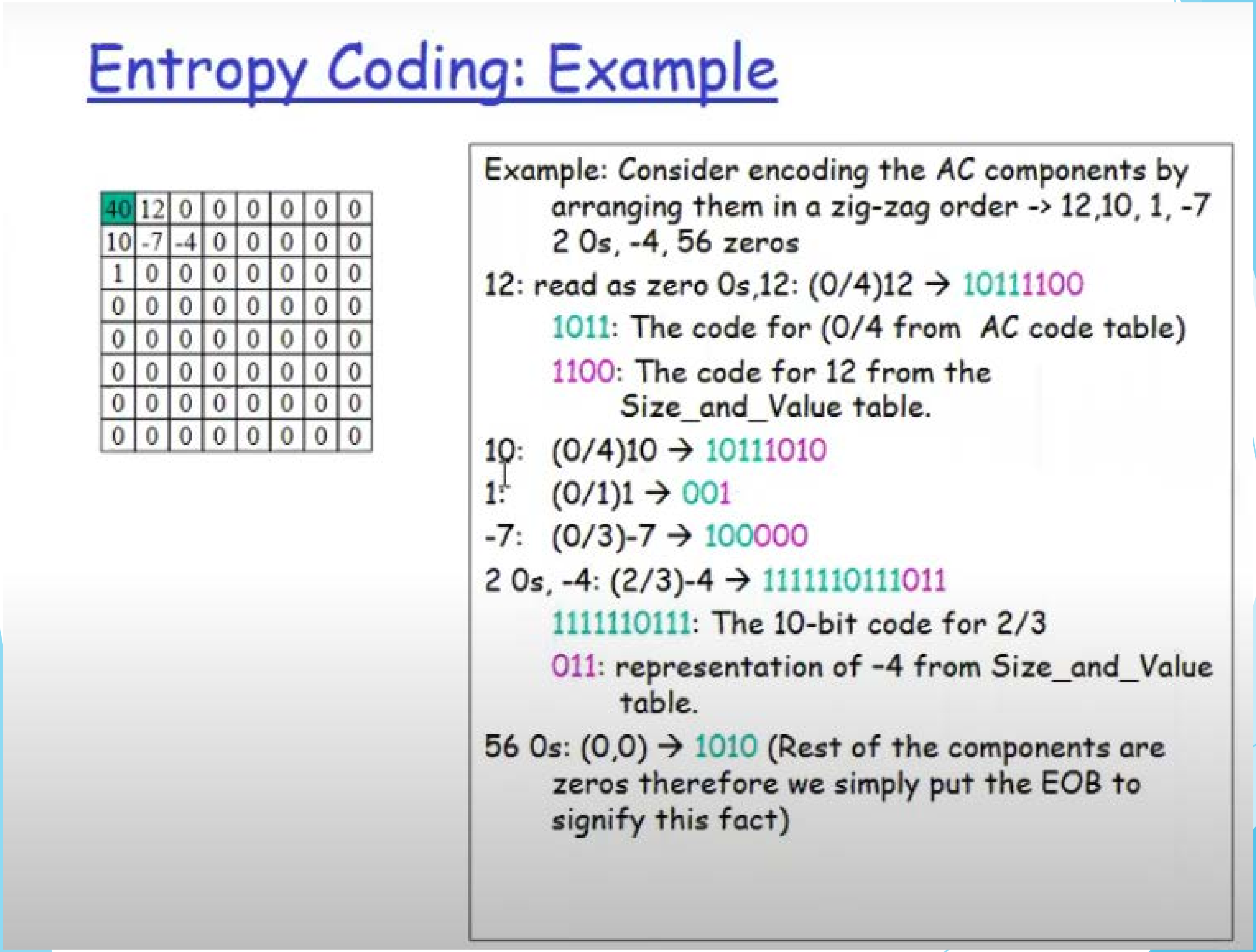
Entropy Coding: AC Components

- AC components are coded as (S1,S2 pairs):
  - **S1: (RunLength/SIZE)**
    - **RunLength:** The length of the consecutive zero values [0..15]
    - **SIZE:** The number of bits needed to code the *next* nonzero AC component's value. [0-A]
    - (0,0) is the End_Of_Block for the 8x8 block.
    - **S1** is Huffman coded (see AC code table below)
  - **S2: (Value)**
    - **Value:** Is the value of the AC component.(refer to size_and_value table)

Partial Huffman Table for AC Run/Size Pairs

Run/ SIZE	Code Length	Code
0/0	4	1010
0/1	2	00
0/2	2	01
0/3	3	100
0/4	4	1011
0/5	5	11010
0/6	7	1111000
0/7	8	11111000
0/8	10	1111110110
0/9	16	1111111110000010
0/A	16	1111111110000011

Run/ SIZE	Code Length	Code
1/1	4	1100
1/2	5	11011
1/3	7	1111001
1/4	9	111110110
1/5	11	11111110110
1/6	16	1111111110000100
1/7	16	1111111110000101
1/8	16	1111111110000110
1/9	16	1111111110000111
1/A	16	1111111110001000
... 15/A	More	Such rows



Entropy Coding: Example

40	12	0	0	0	0	0	0
10	-7	-4	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
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Example: Consider encoding the AC components by arranging them in a zig-zag order -> 12,10, 1, -7 2 0s, -4, 56 zeros

12: read as zero 0s,12: (0/4)12 → 10111100

1011: The code for (0/4 from AC code table)

1100: The code for 12 from the Size_and_Value table.

10: (0/4)10 → 10111010

1: (0/1)1 → 001

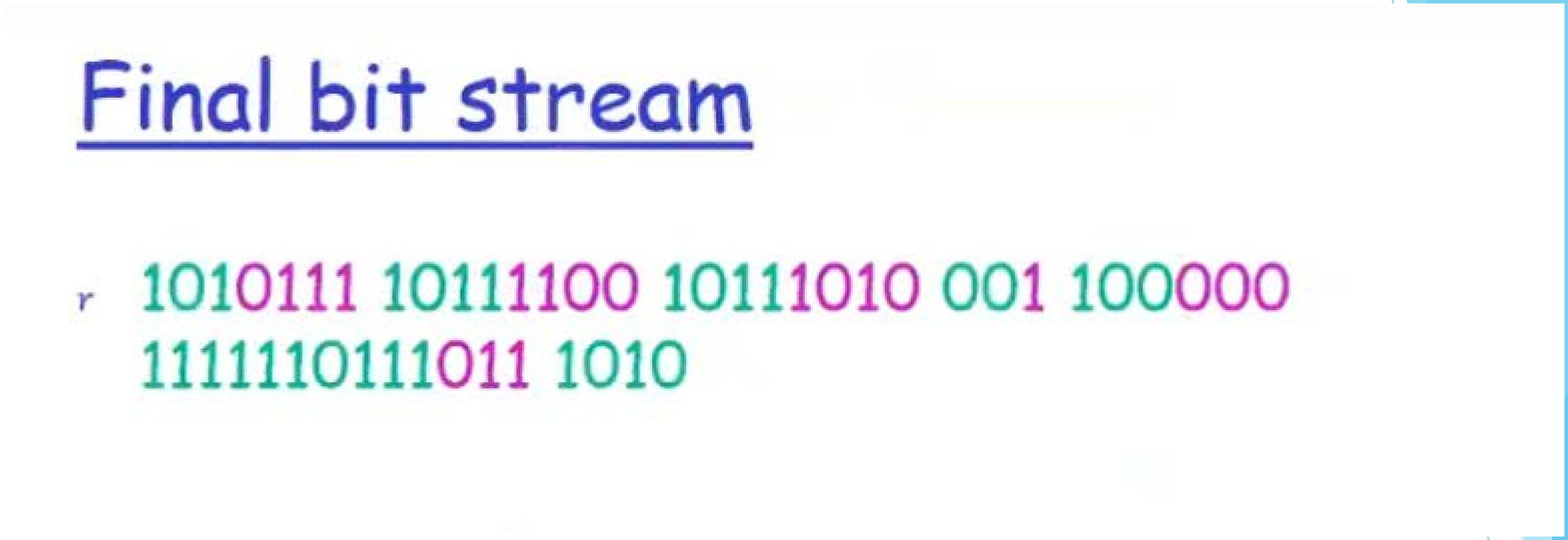
-7: (0/3)-7 → 100000

2 0s, -4: (2/3)-4 → 1111110111011

1111110111: The 10-bit code for 2/3

011: representation of -4 from Size_and_Value table.

56 0s: (0,0) → 1010 (Rest of the components are zeros therefore we simply put the EOB to signify this fact)



Final bit stream

- 1010111 10111100 10111010 001 100000 1111110111011 1010

JPEG Extensions

- The basic mode of JPEG supports sequential coding (the order is from top to bottom and left to right).
- JPEG extensions support progressive modes:
  - Spectrum selection.
  - Successive approximation.
  - Hierarchical mode.

JPEG 2000

- JPEG 2000 is a new standard based on wavelet transform.
- JPEG2000 does not partition an image into small blocks.
- Wavelet decomposes the whole image into different “bands” and then encodes the coefficients in different bands smartly.

Thank You