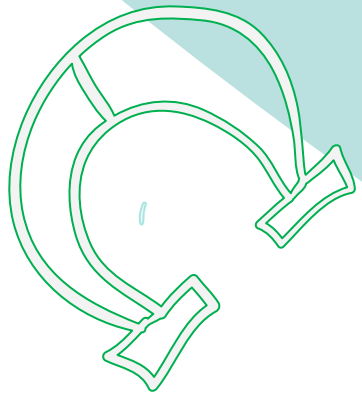
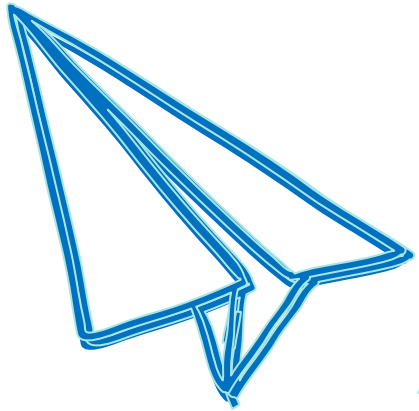
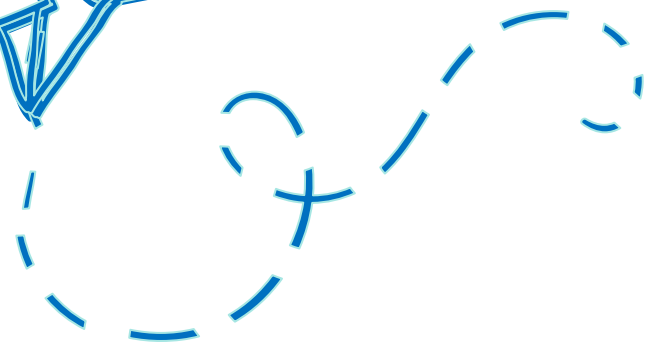
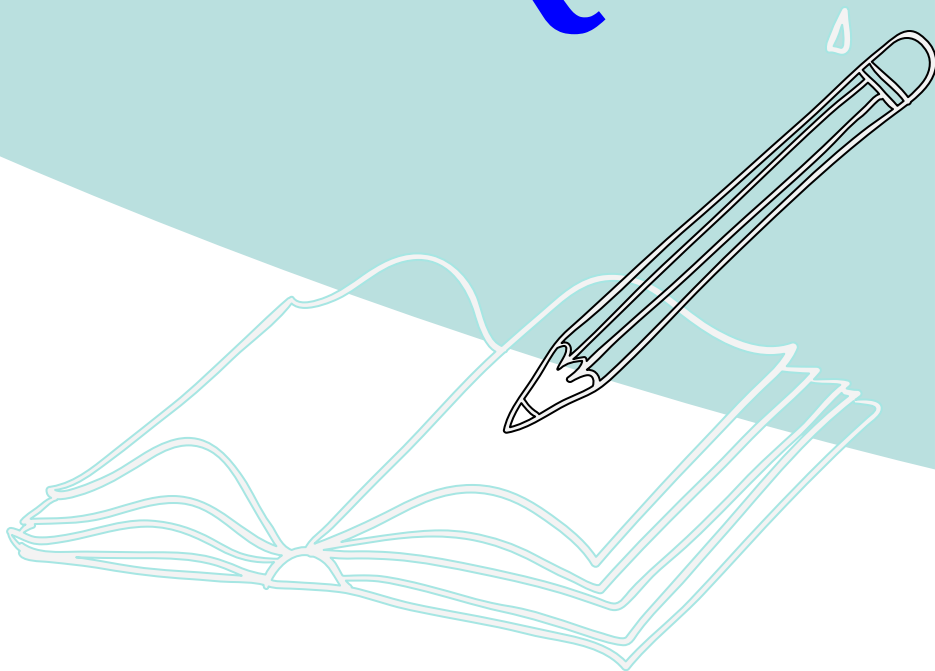


NÉN ẢNH BẰNG PHƯƠNG PHÁP WSQ

(PYTHON)







WAVELET SCALAR QUANTIZATION

1. **Pre-processing & Post-processing**
2. Wavelet Transformation & IDWT
3. Quantization & Dequantization
4. Huffman encoding & Decoding

PROCESSING CYCLE

- Converting to grayscale
- Scaling to x
- I’ =
- I = I’.s + m
- Scaling to initial size
- Reconstructing the color image

Explanation:

- Converting from color images to grayscale
- We scale the size of the input image to the given size in order to meet the requirement of the problem
- The provide formula ensures that roughly half of the new pixel values are negative, while the other half are positive, and all fall in the range we want. To be specific, given:

I: the original image

m: the mean pixel value

$$s = \frac{\max(\max(I) - m, m - \min(I))}{s'}$$

Note: s' which decides the range depends on the bit values giving the grayscale pixel values (eg: input is a matrix of nonnegative 8-bit integer values)

WAVELET SCALAR QUANTIZATION

1. Pre-processing & Post-processing
2. **Wavelet Transformation & IDWT**
3. Quantization & Dequantization
4. Huffman encoding & Decoding

Fourier vs wavelet

- **Fourier Transformation converts a signal from time domain to frequency domain**. However, there exists a drawback which is that in terms of a signal f(t), we **do not know that the frequency components of f(t) at an instant moment t.** As a consequence, a better transformation is supposed to not only possess all the advantages of Fourier Transformation, but it also possibly define the frequency components of a given signal f(t) at an instant moment t. The advent of Wavelet Transformation has been able to cover the drawback of Fourier Transformation.
- **In 1975, Jean Morlet developed the Multiresolution Analysis**. He used an oscillation impulse, known as "wavelet" (small wave) – enable to change the size, to compare with the signal in different bands. At the beginning, the wavelet started with low-frequency oscillation. After that, the wavelet was then compressed so as to increase the frequency of the oscillation. This process is called changing the analyzing scale, which helps to the **detect the sharply fluctuating hidden signal** – the initial purpose of Wavelet Transformation.

What's wrong with Fourier?

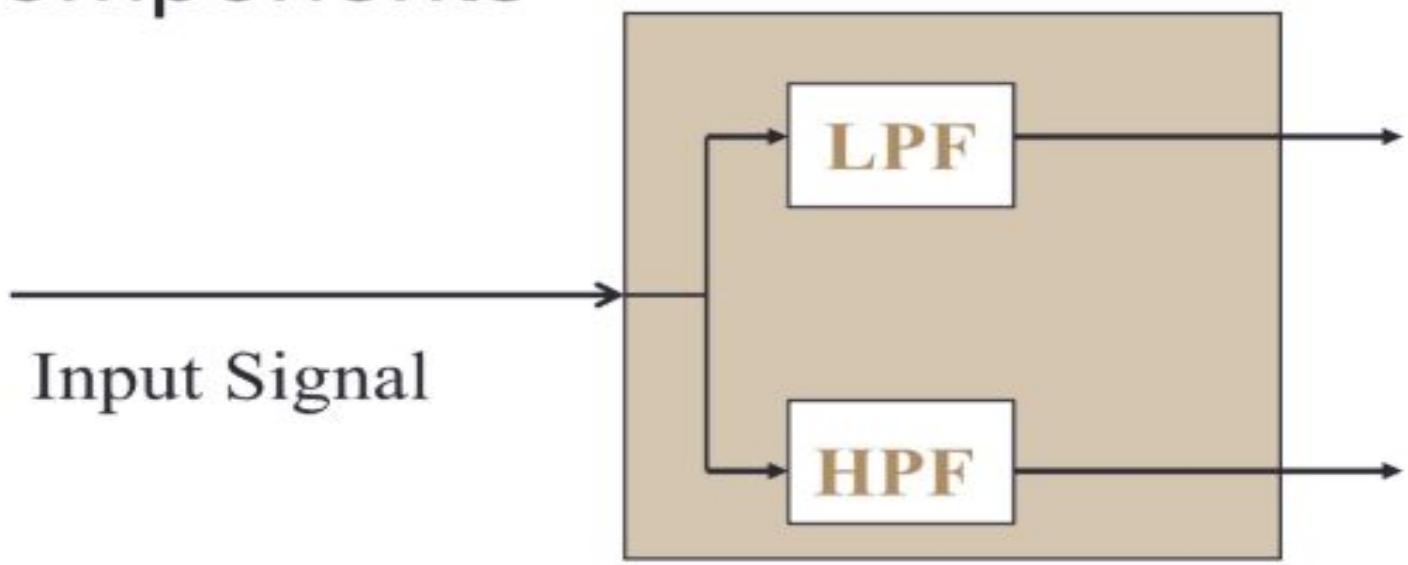
- By using Fourier Transform , we loose the time information : WHEN did a particular event take place ?
- FT can not locate drift, trends, abrupt changes, beginning and ends of events, etc.
- Calculating use complex numbers.

Multi Resolution Analysis – MRA

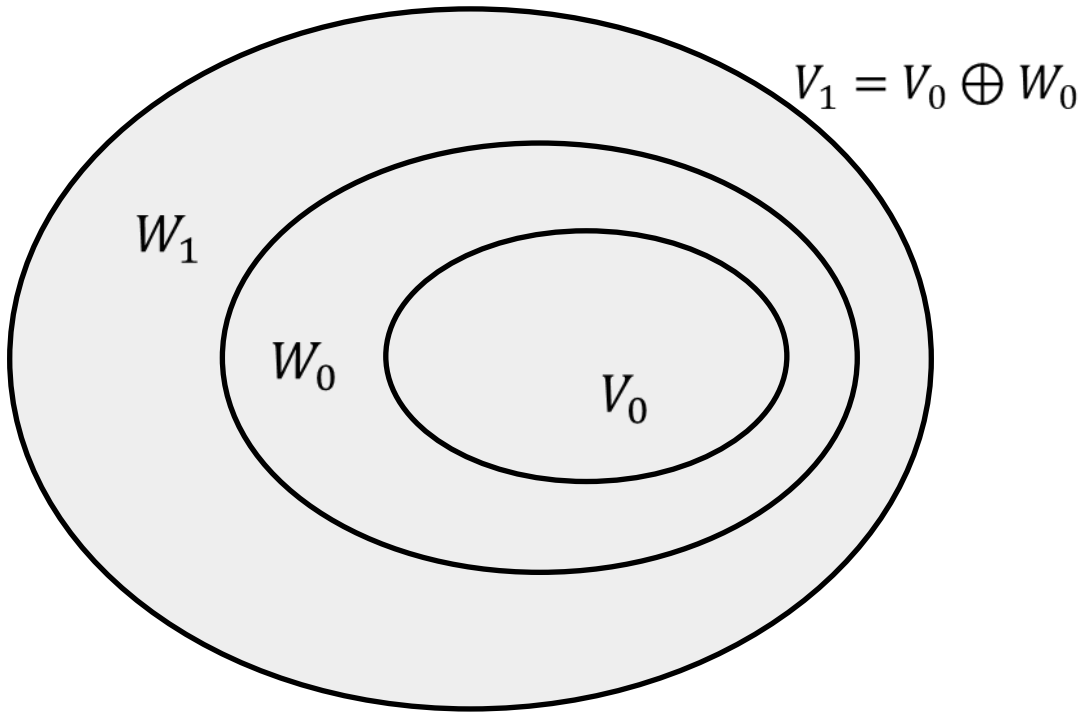
- Phân tích đa phân giải (Multi Resolution Analysis – MRA) phân tích tín hiệu ra các dải tần số khác nhau thông qua các bộ lọc thông thấp và bộ lọc thông cao liên tiếp.
- MRA có khả năng như hai bộ lọc để tạo ra hai thành phần chi tiết và xấp xỉ.
- Thành phần chi tiết có hệ số tỷ lệ thấp tương ứng với thành phần thành phần tần số cao được thực hiện thông qua bộ lọc thông cao, thành phần xấp xỉ có hệ số tỷ lệ cao tương ứng với thành phần tần số thấp được thực hiện thông qua bộ lọc thông thấp.

Approximations and Details:

- Approximations: High-scale, low-frequency components of the signal
- Details: low-scale, high-frequency components



$$V_2 = V_1 \oplus W_1 = V_0 \oplus W_0 \oplus W_1$$



The relationship between scaling and wavelet fuction spaces

In the translating example, we found a fuction wich is orthonormal to its transling version. Different scaled version can see different frequency resolutions. Combined with the 2 properties, we can canstruct a basis from the scaling fuction and wavelet function with 2 parameters: scaling and translating, formally defined in pp. 365 – 372 [1] as

$$\emptyset_{j,k}(t) = 2^{\frac{j}{2}}\emptyset\left(2^{\mathrm{j}}\mathrm{t} - \mathrm{k}\right)$$

$$\psi_{j,k}(t) = 2^{\frac{j}{2}}\psi(2^{\mathrm{j}}\mathrm{t} - \mathrm{k})$$

Where j is the parameter about dilation, or the visibility in frequency and k is parameter about the position. In practice, we may want to see the whole data with “desired” resolution, i.e, for some resolution j. We define the subspace

$$V_j = Span\{\phi_{j,k}(t)\}$$

$$W_j = Span\{\psi_{j,k}(t)\}$$

$$f(n) = \frac{1}{\sqrt{M}} \sum_{k} W_{\phi}[j_0, k]\phi_{j_0,k}[n] + \frac{1}{\sqrt{M}} \sum_{j=j_0}^{\infty} \sum_{k} W_{\psi}[j, k]\psi_{j,k}[n]$$

Here $f(n), \phi_{j_0,k}[n]$ *and* $\psi_{j,k}[n]$ are discrete functions defined in [0, M – 1], totally M point. Because the sets $\{\phi_{j_0,k}[n]\}_{k\epsilon\mathbb{Z}}$ and $\{\psi_{j,k}[n]\}_{(j,k)\epsilon\mathbb{Z}^2, j\geq j_0}$ are orthogonal to each other. We can simply take the inner product to obtain the wavelet coefficients

$$W_{\phi}[j_0, k] = \frac{1}{\sqrt{M}} \sum_{n} f(n)\phi_{j_0,k}[n] \qquad (*)$$

$$W_{\psi}[j, k] = \frac{1}{\sqrt{M}} \sum_{n} f(n)\psi_{j,k}[n] \qquad (**)$$

(*) are called approximation coeficients while (**) are called detailed coefficients

$$X_w(a,b) = \frac{1}{|a|^{1/2}} \int_{-\infty}^{\infty} x(t)\overline{\psi}\left(\frac{t-b}{a}\right) dt$$

$$x(t) = C_{\psi}^{-1} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} X_w(a,b) \frac{1}{|a|^{1/2}} \tilde{\psi}\left(\frac{t-b}{a}\right) db \frac{da}{a^2}$$

$\tilde{\psi}(t)$ is the dual function of $\psi(t)$ and

$$C_{\psi} = \int_{-\infty}^{\infty} \frac{\overline{\hat{\psi}}(\omega)\hat{\tilde{\psi}}(\omega)}{|\omega|} d\omega$$

Continuous Wavelet transformation

In mathematics, the continuous wavelet transform (CWT) is a formal tool that provides an overcomplete representation of a signal by letting the translation and scale parameter of the wavelets vary continuously

Continuous Wavelet transformation

$$CWT(\tau,\alpha) = \frac{1}{\sqrt{|\alpha|}} \int_{-\infty}^{\infty} f(t) g * \left(\frac{t-\tau}{\alpha}\right) e^{-2\pi i k_0 \frac{t-\tau}{\alpha}} dt$$

This is just here for weighting

Note that as alpha increases: the frequency decreases, and the widow function expands

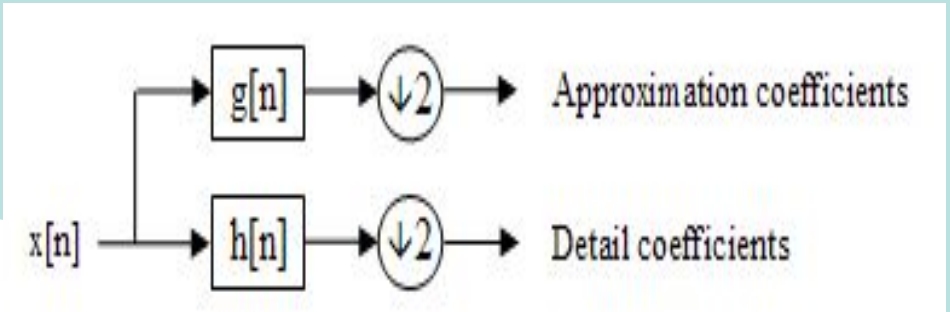
- *tau: shift coefficient*
- *alpha: scale coefficient*

Discrete Wavelet transformation

- A discrete wavelet transform (DWT) is any wavelet transform for which the wavelets are discretely sampled.
- The DWT of a signal x is calculated by passing it through a series of filters. First the samples are passed through a low pass filter with impulse response g resulting in a convolution of the two:

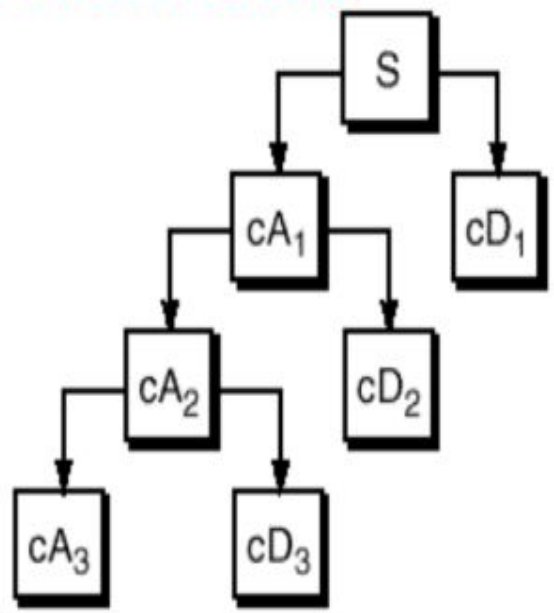
$$y[n] = (x * g)[n] = \sum_{k=-\infty}^{\infty} x(k).g(n - k)$$

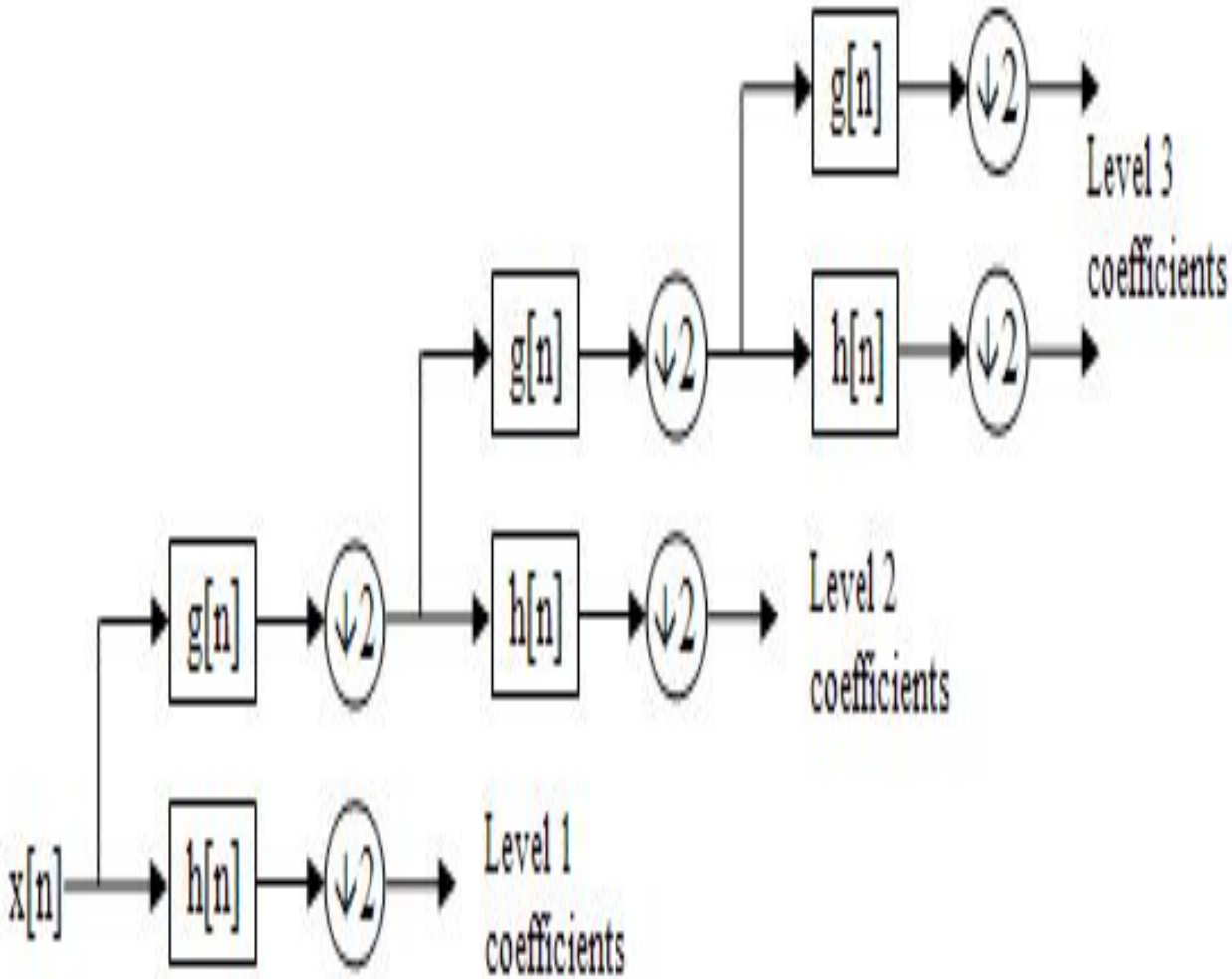
- The signal is also decomposed simultaneously using a high-pass filter h. The outputs give the detail coefficients (from the high-pass filter) and approximation coefficients (from the low-pass). It is important that the two filters are related to each other and they are known as a quadrature mirror filter



Multi-level Decomposition

- Iterating the decomposition process, breaks the input signal into many lower-resolution components: *Wavelet decomposition tree*:




x[n]
h[n]
g[n]
↓2
Level 1
coefficients
h[n]
Level 2
coefficients
h[n]
Level 3
coefficients

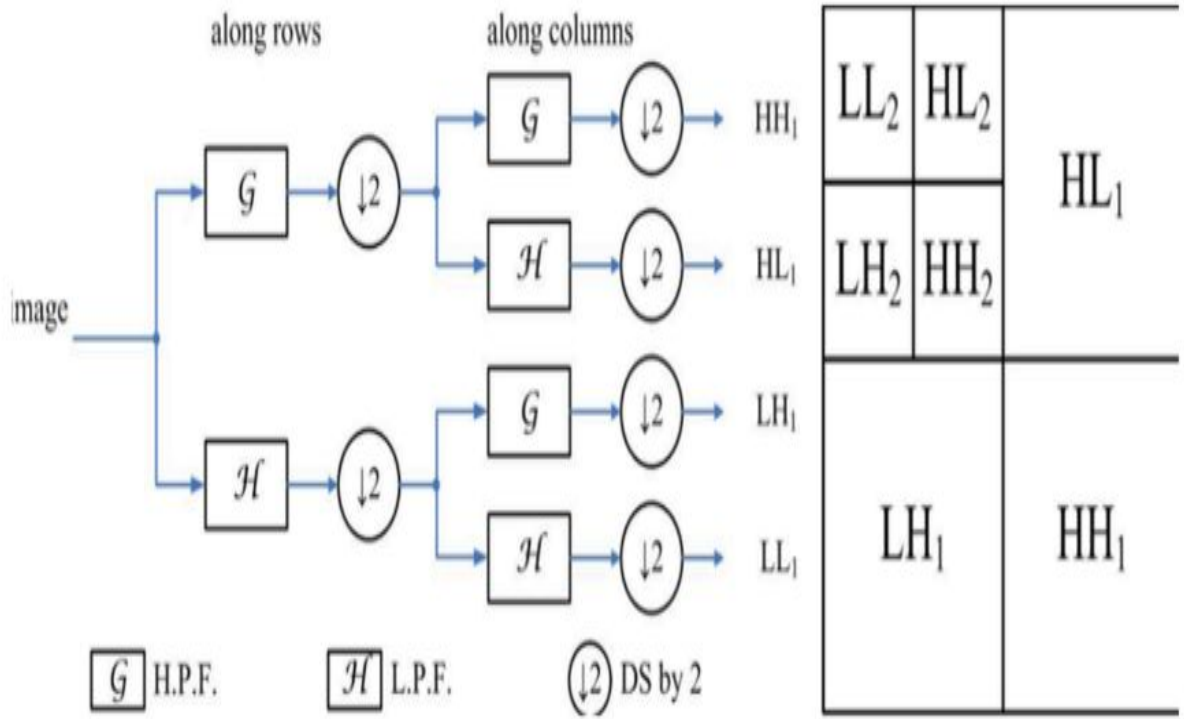
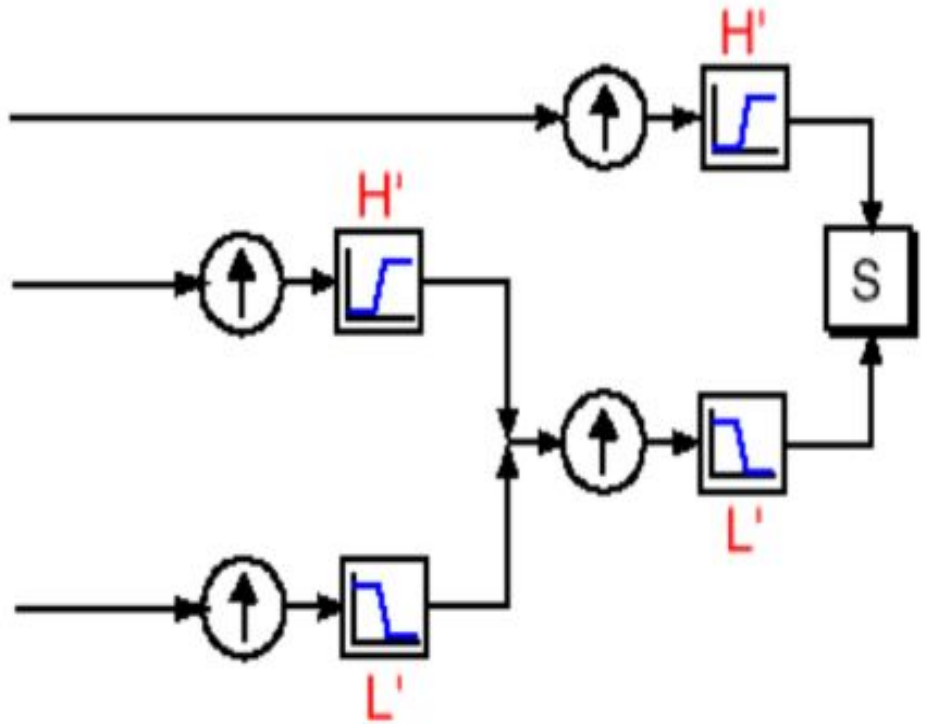


image
along rows
along columns
G
H
↓2
HH1
HL1
LH1
LL1
LL2
HL2
LH2
HH2
HL1
LH1
HH1
G H.P.F.
H L.P.F.
↓2 DS by 2

Wavelet reconstruction

- Reconstruction (or synthesis) is the process in which we assemble all components back



Up sampling (or interpolation) is done by zero inserting between every two coefficients

WAVELET SCALAR QUANTIZATION

1. Pre-processing & Post-processing
2. Wavelet Transformation & IDWT
3. **Quantization & Dequantization**
4. Huffman encoding & Decoding

Scalar Quantization

- **Quantization is the process of mapping each wavelet coefficient to an integer value**, and is the main source of compression in the algorithm. By mapping the wavelet coefficients to a relatively small set of integer values, the complexity of the data is reduced, which allows for efficient encoding of the information in a bit string. **Further, a large portion of the wavelet coefficients will be mapped to 0 and discarded completely**. Care must be taken, however, to perform this quantization in a manner that achieves good compression without discarding so much information that the image cannot be accurately reconstructed.
- **The values Z_k and Q_k are dependent on the subband and determine how much compression is achieved. If $Q_k = 0$, all coefficients are mapped to 0**. Selecting appropriate values for these parameters is a tricky problem in itself and relies on heuristics based on the statistical properties of the wavelet coefficients. Therefore, the methods that calculate these values have already been initialized. **Quantization is not a perfectly invertible process. Once the wavelet coefficients have been quantized, some information is permanently lost**. However, wavelet coefficients a^k in subband k can be roughly reconstructed from the quantized coefficients p using the following formula. This

Fig(7) shows the setup of quantization bins for sub band "K". Parameter Z_k is the width of the zero bin, and parameter Q_k is the width of the other bins. Parameter "C" is the range [0,1]. It determines the reconstructed value $'\hat{a}'$. For C=0.5, the reconstructed value for each quantization bin is the center of the bin[4,7].

$$P_k^{(m,n)} = \begin{cases} \left[\dfrac{a_k^{(m,n)} - Z_{k/2}}{Q_k}\right] + 1, & a_k^{(m,n)} > Z_{k/2} \\ 0 & -Z_{k/2} \le a_k(m,n) \le Z_{k/2} \\ \left[\dfrac{a_k^{(m,n)} + Z_{k/2}}{Q_k}\right] + 1, & a_k^{(m,n)} < -Z_{k/2} \end{cases} \quad \ldots\ldots(1)$$

$$'\hat{a}' = \begin{cases} (P_k^{(m,n)} - C)Q_k + Z_{k/2}, & P_k^{(m,n)} > 0 \\ 0 & P_k^{(m,n)} = 0 \\ (P_k^{(m,n)} + C)Q_k - Z_{k/2}, & P_k^{(m,n)} < 0 \end{cases} \quad \ldots\ldots(2)$$

- The transform coefficients in the 64 sub-bands are floating-point numbers to be denoted by 'a'. They are quantized to a finite number of floating-point numbers that are denoted by 'a^'. The encoder maps a transform coefficient 'a' to a quantization index 'p'. The index 'p' can be considered a pointer to the quantization bin where 'a' lies.
- The decoder receives an index "p' and maps it to a value 'a^' that is close, but not identical, to 'a'. The set of 'a^' values is a discrete set of floating-point numbers called the quantized wavelet coefficients. The quantization depends on parameters that may vary from sub-band to sub-band, since different sub-band have different quantization requirements.

WAVELET SCALAR QUANTIZATION

1. Pre-processing & Post-processing
2. Wavelet Transformation & IDWT
3. Quantization & Dequantization
4. **Huffman encoding & Decoding**

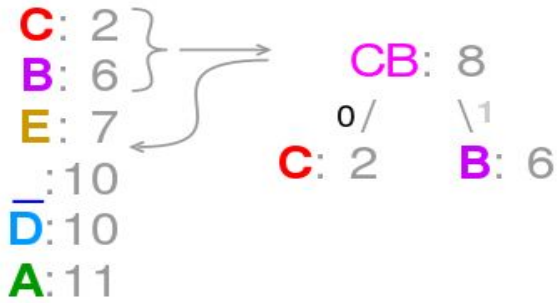
Grouping and Huffman encoding

- A Huffman code is a **particular type of optimal prefix code that is commonly used for lossless data compression**. The process of finding or using such a code proceeds by means of Huffman coding, an algorithm developed by David A. Huffman while he was a Sc.D. student at MIT and published in the 1952 paper "A Method for the Construction of Minimum-Redundancy Codes"
- **Prefix Codes, means the codes (bit sequences) are assigned in such a way that the code assigned to one character is not the prefix of code assigned to any other character**. This is how Huffman Coding makes sure that there is no ambiguity when decoding the generated bitstream.
- Let us understand prefix codes with a counter example. Let there be four characters a, b, c and d, and their corresponding variable length codes be 00, 01, 0 and 1. This coding leads to ambiguity because code assigned to c is the prefix of codes assigned to a and b. If the compressed bit stream is 0001, the de-compressed output may be “cccd” or “ccb” or “acd” or “ab”
- **Once all of the subbands have been quantized, they are divided into three groups**. The first group contains the smallest ten subbands (positions zero through nine), while the next two groups contain the three subbands of next largest size (positions ten through twelve and thirteen through fifteen, respectively). **All of the subbands of each group are then flattened and concatenated with the other subbands in the group. These three arrays of values are then mapped to Huffman indices.**

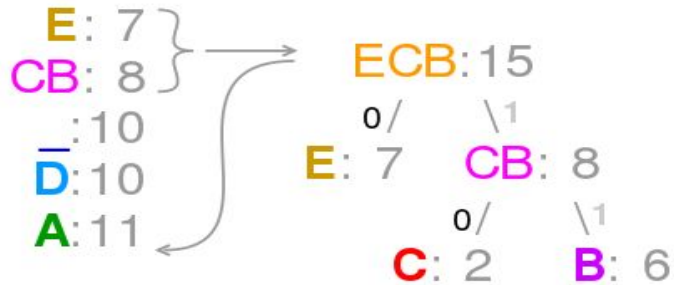
This allows large chunks of information to be stored as a single index, greatly aiding in compression. The Huffman indices are then assigned a bit string representation through a

1. "A_DEAD_DAD_CEDED_A_BAD_BABE_A_BEADED_ABACA_BED"

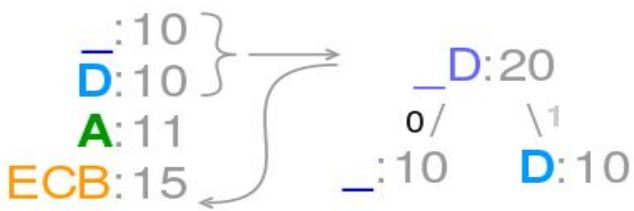
2.



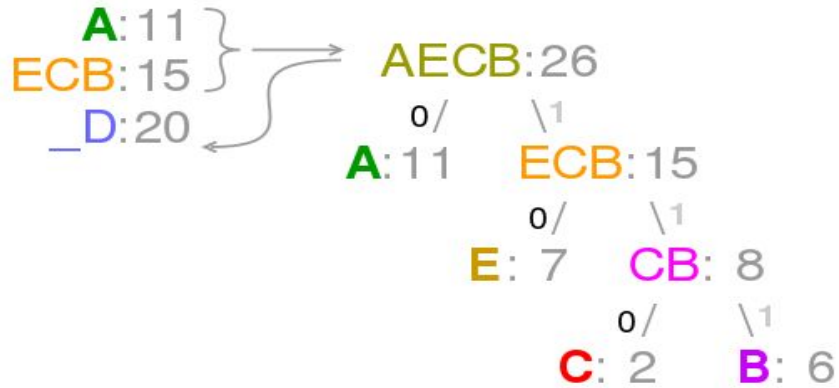
3.



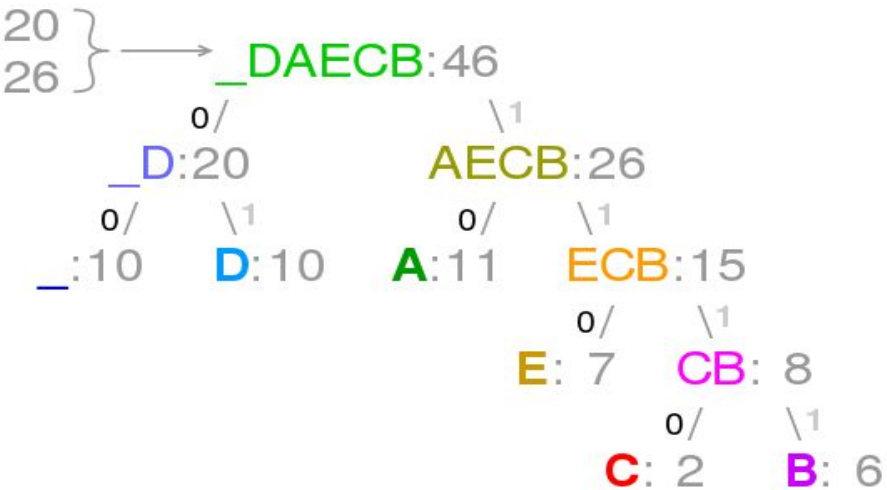
4.



5.



6.



7.

_: 00
D: 01
A: 10
E: 110
C: 1110
B: 1111

8. "1000011101001000110010011101100111001001000111111001001111110111111000100011111110100111001001011111011101000111111001"

** Peak signal to noise ratio (PSNR) is calculated as follows

$$PSNR = 20 \log_{10} \frac{\max_i / pi /}{RMSE}$$

* Compression Ratio (CR) is calculated as follows

CR = N1/N2

N1–number of information carrying bytes in the original image.

N2–number of information carrying bytes in the encoded image.

Root mean square error (RMSE) =

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(pi - Qi)^2}$$

Where Pi = pixel of original image, Qi = pixel of reconstructed image. Here the encoded image is MAT file.

Evaluation of reconstructed image

- Peak signal to noise ratio
- This ratio is used as a quality measurement between the original and a compressed image. **The higher the PSNR, the better the quality of the compressed, or reconstructed image.**
- Compression ratio
- Root mean square error

PSNR & MSE

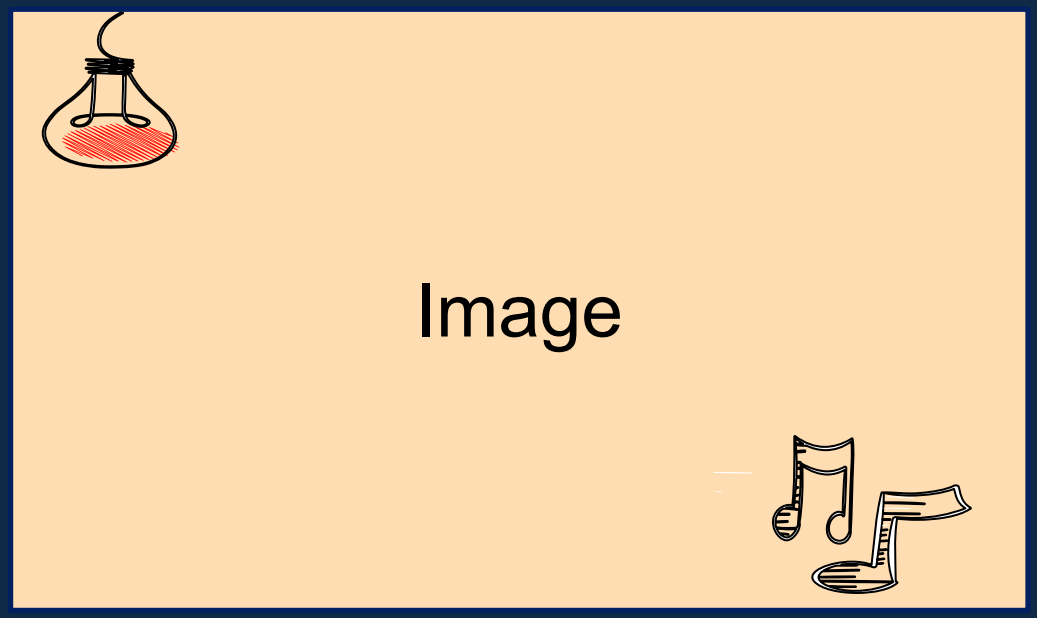
To compute the PSNR, the block first calculates the mean-squared error using the following equation

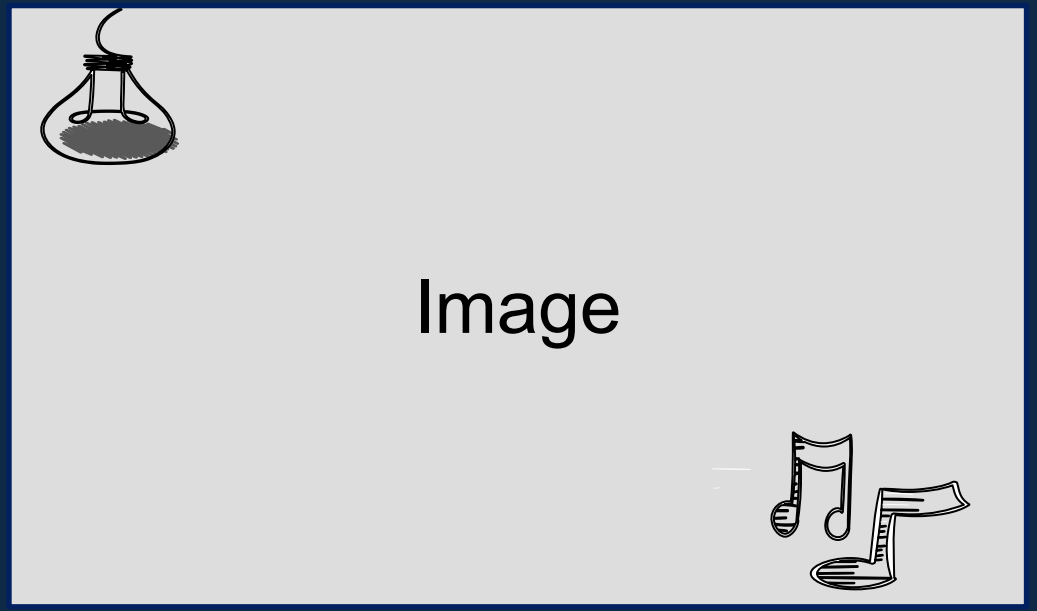
$$MSE = \frac{\sum_{M,N}[l_1(m,n) - l_2(m,n)]^2}{M * N}$$

In the previous equation, M and N are the number of rows and columns in the input images. Then the block computes the PSNR using the following equation

$$PSNR = 10log_{10}(\frac{R^2}{MSE})$$

IMAGE GRAY





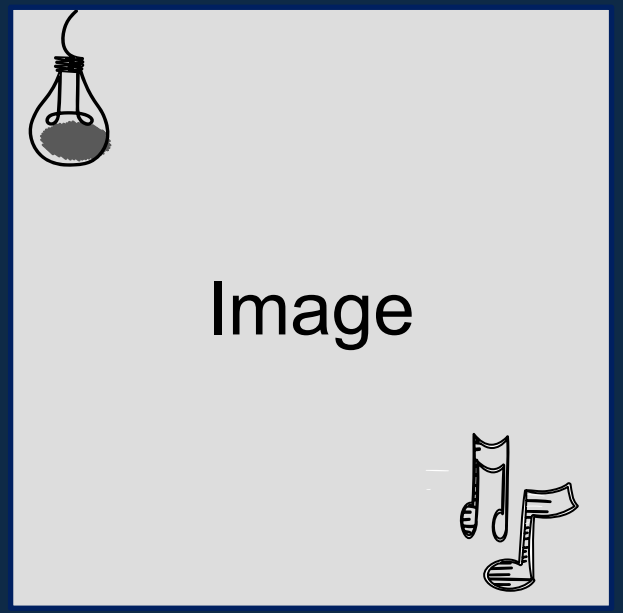
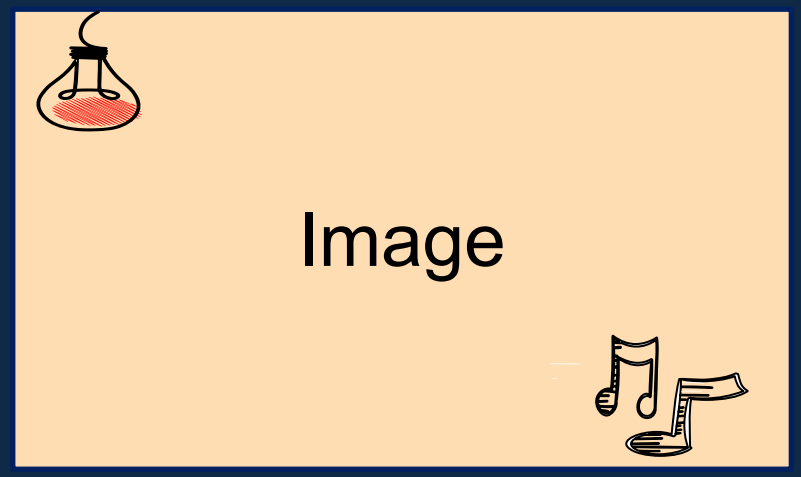
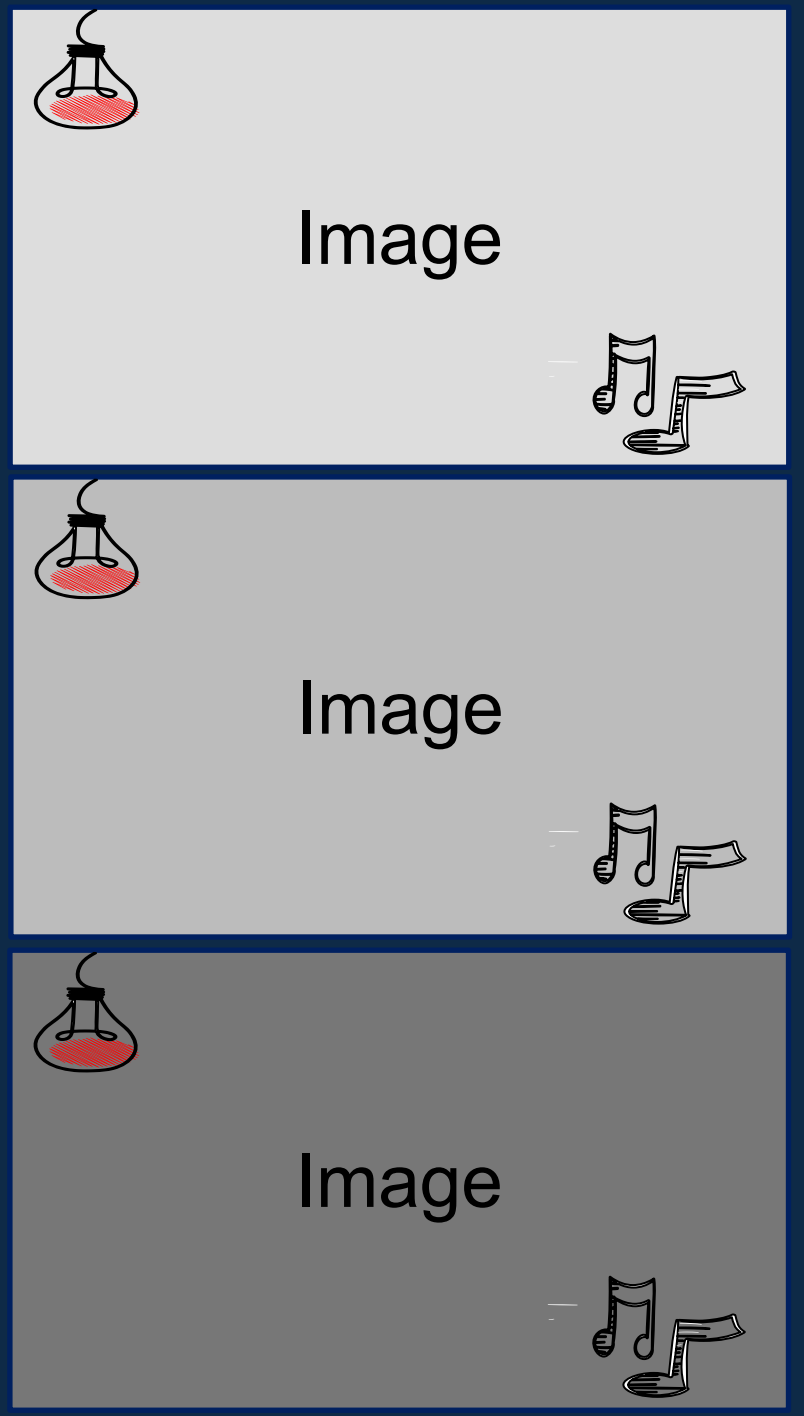
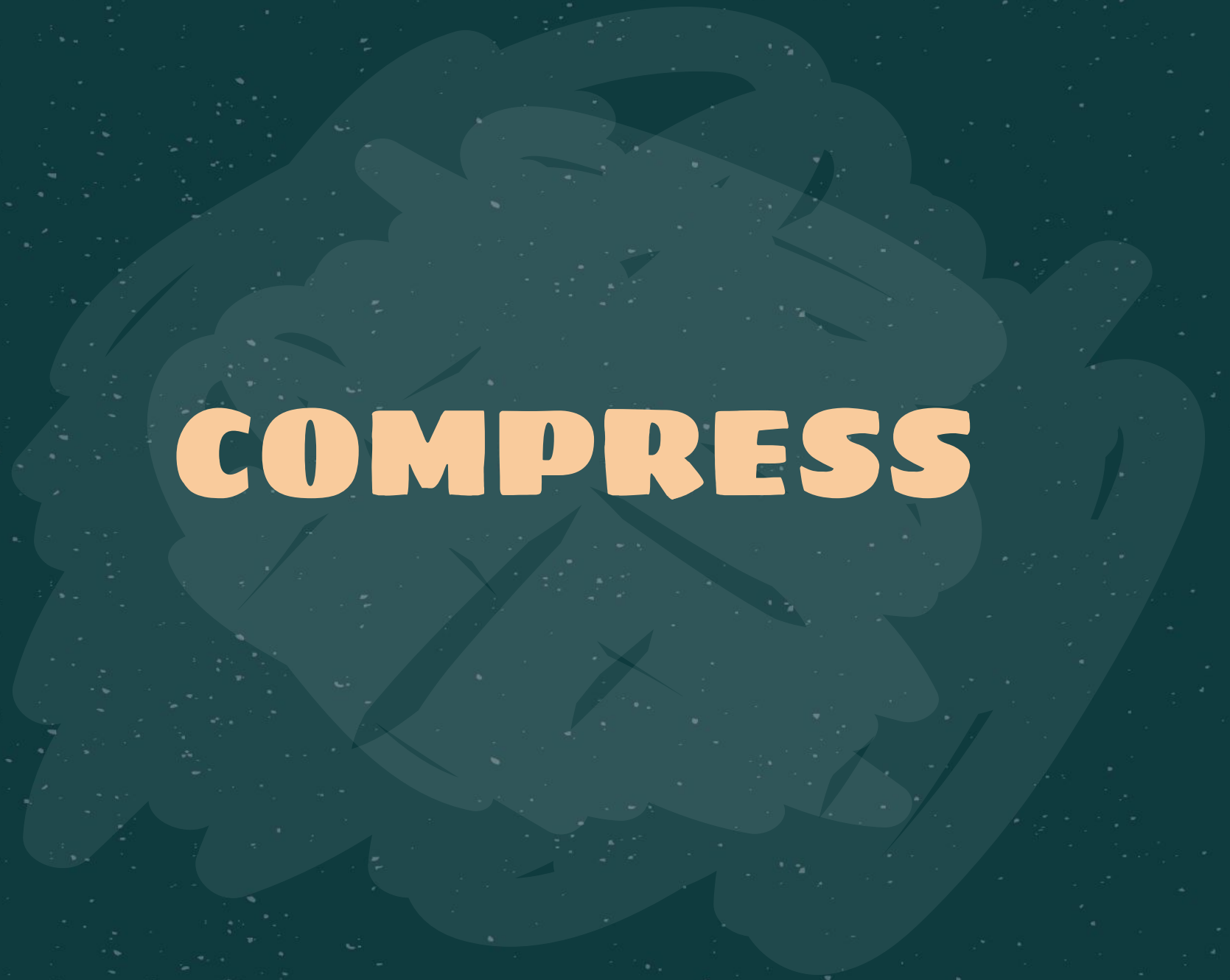


IMAGE COLOR






COMPRESS

Decompose

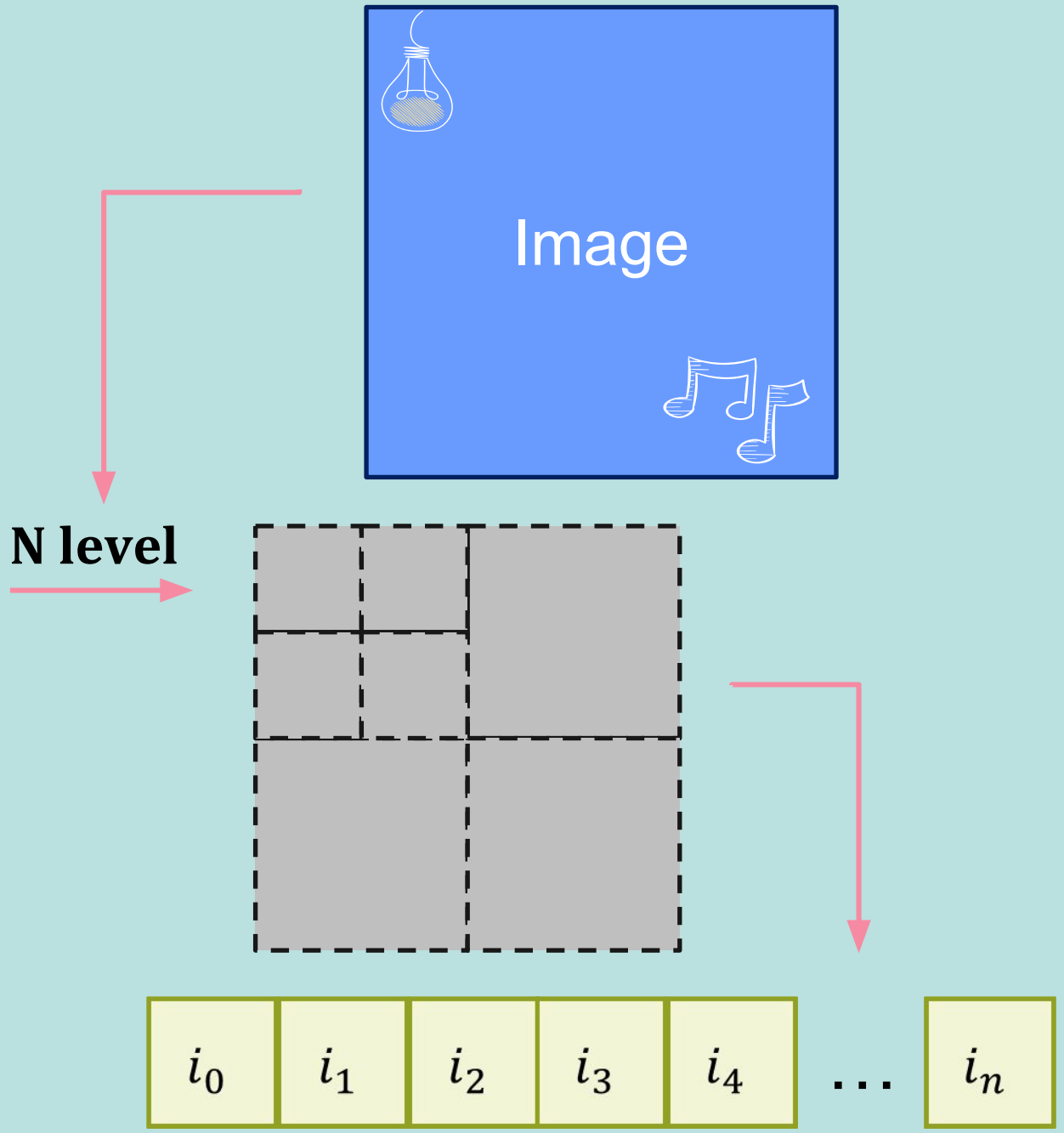
Get_bin

Quantize

Group

Encode

Bitstrings



Decompose

Get_bin

Quantize

Group

Encode

Bitstrings

SUBBANDS

$$x_k = \left\lfloor \frac{3X_k}{4} \right\rfloor \; and \; y_k = \left\lfloor \frac{7X_k}{16} \right\rfloor$$

$$x = \left\lfloor \frac{X_k}{8} \right\rfloor \; and \; y = \left\lfloor \frac{9X_k}{32} \right\rfloor$$

$$\sigma_k^2 = \frac{1}{X_k' Y_k' - 1} \sum_{n=x_{0,k}}^{x_{1,k}} \sum_{m=y_{0,k}}^{y_{1,k}} (a_k(m,n) - \mu_k)^2$$

Q, Z

Decompose

Get_bin

Quantize

Group

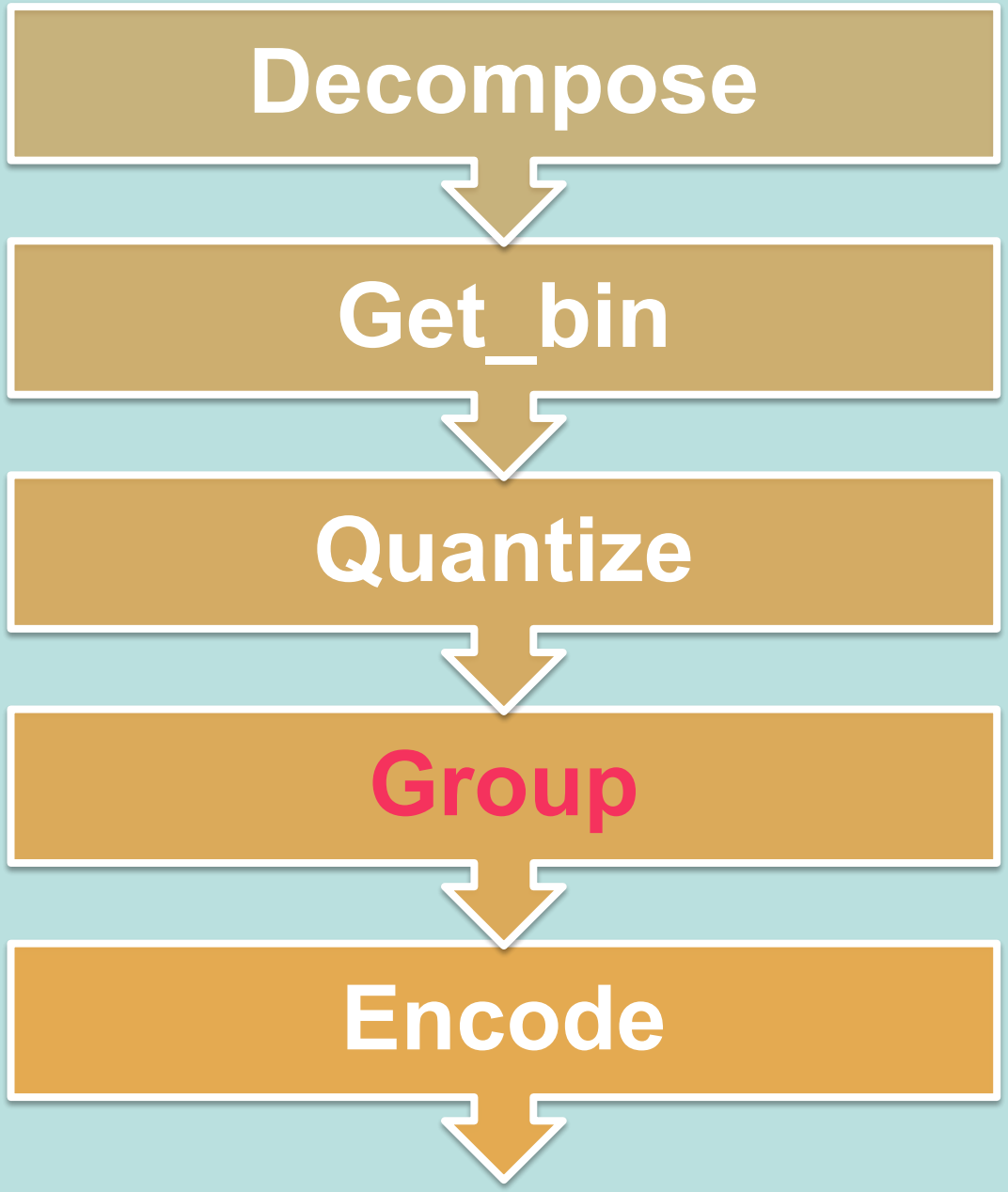
Encode

Bitstrings

SUBBANDS

$$p_k(m,n) = \begin{cases} \left\lfloor \dfrac{a_k(m,n) - \frac{Z_k}{2}}{Q_k} \right\rfloor + 1, & a_k(m,n) > \dfrac{Z_k}{2} \\ 0, & \dfrac{-Z_k}{2} \le a_k(m,n) \le \dfrac{Z_k}{2} \\ \left\lfloor \dfrac{a_k(m,n) + \frac{Z_k}{2}}{Q_k} \right\rfloor - 1, & a_k(m,n) < \dfrac{-Z_k}{2} \end{cases}$$

SUBBANDSQ



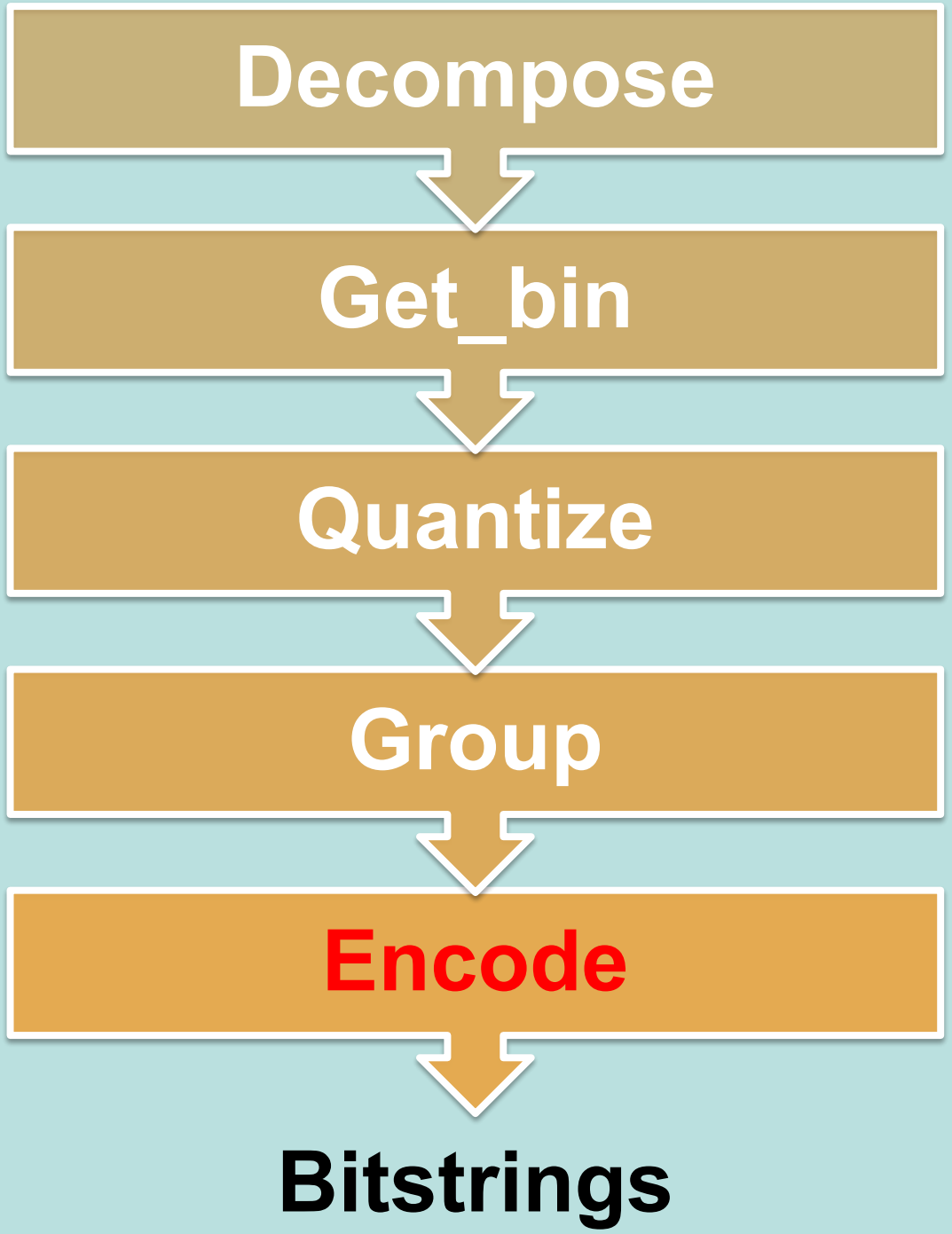
$$shape(image) = mxn$$

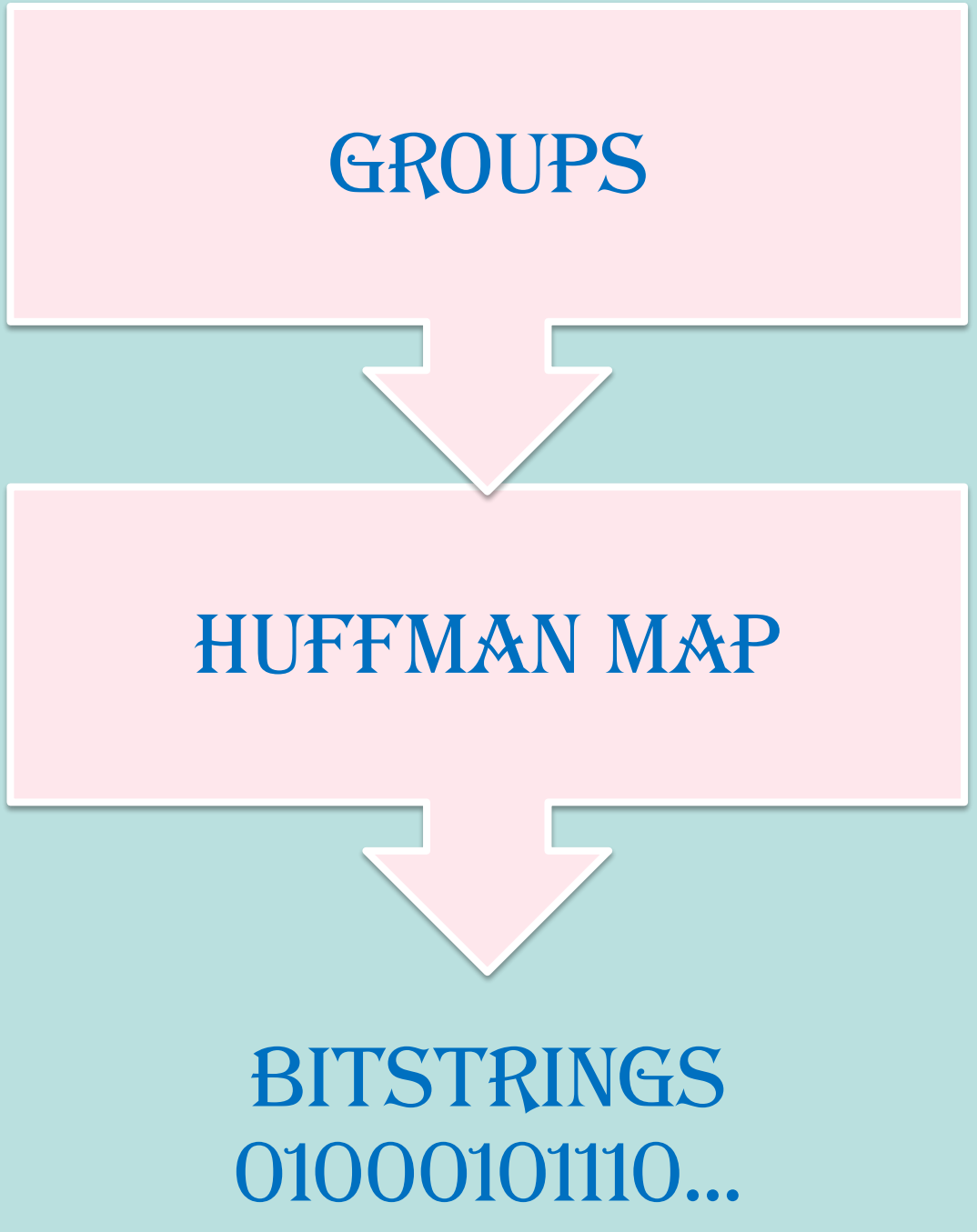
SUBBANDSQ

3 group, with shape(i, j)

$$\begin{cases} other \\ i = \frac{m}{4}, j = \frac{n}{4} \\ i = \frac{m}{2}, j = \frac{n}{2} \end{cases}$$

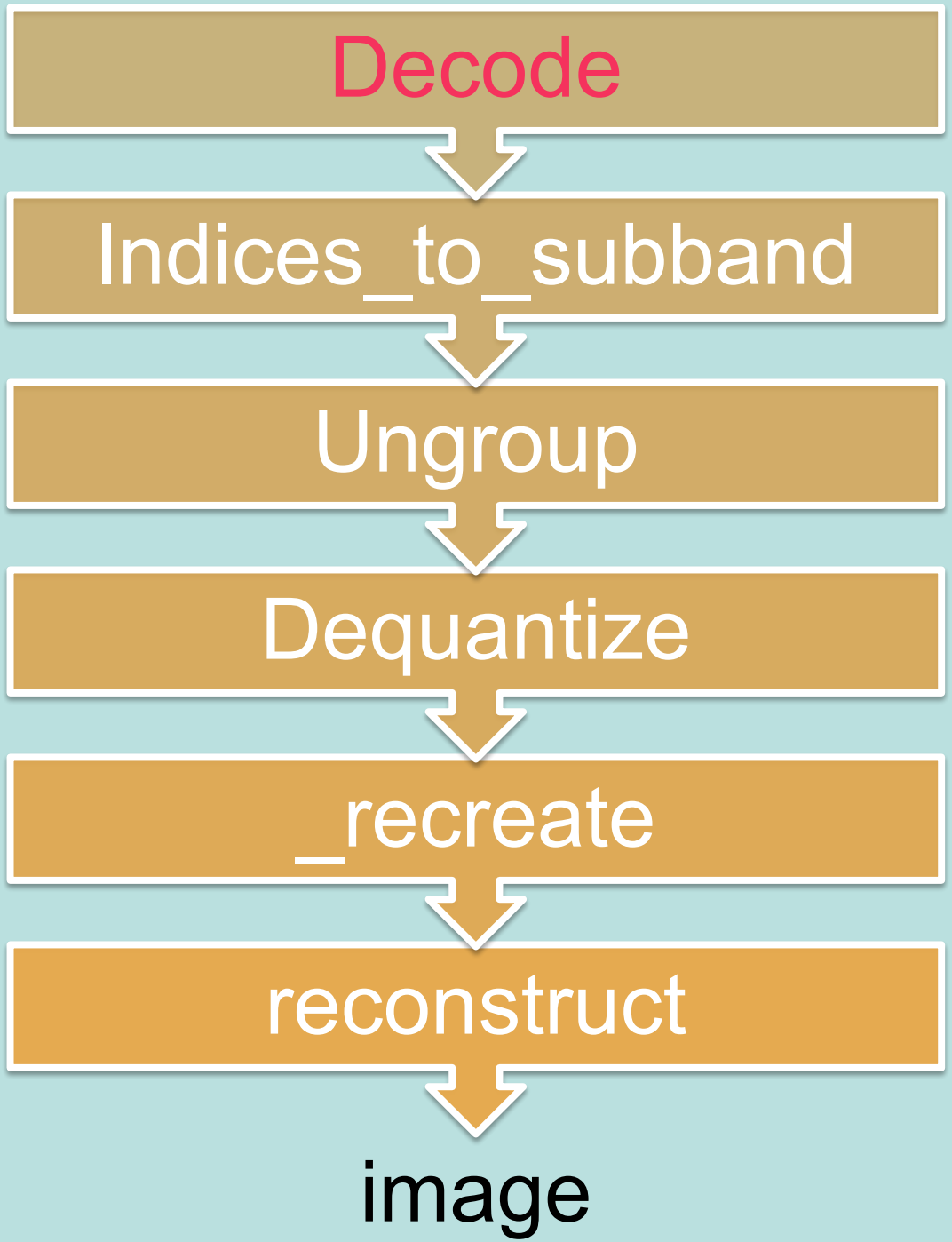
(gs1, gs2, gs3), (ss1, ss2, ss3), (ts1, ts2, ts3)

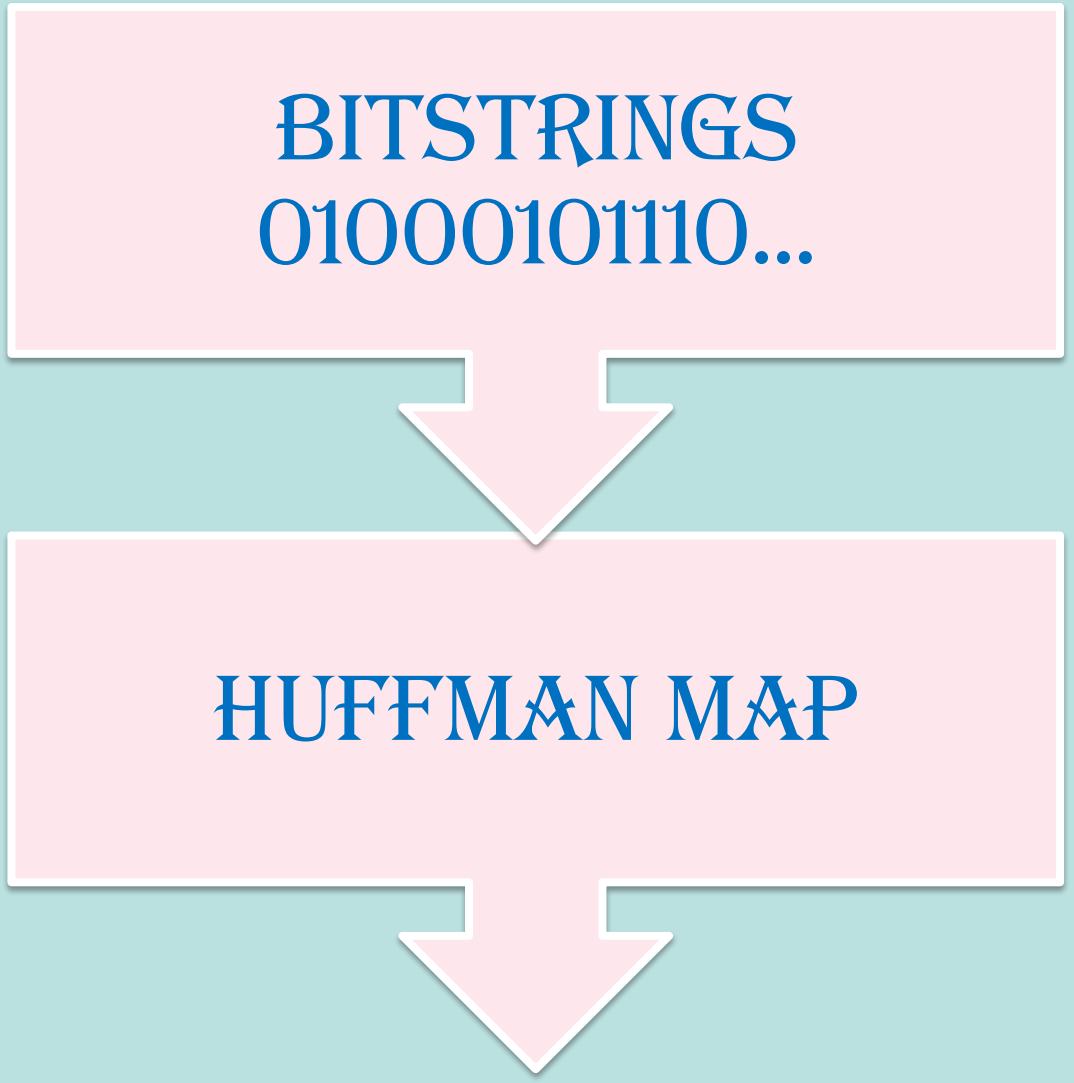

Decompose
Get_bin
Quantize
Group
Encode
Bitstrings


GROUPS
HUFFMAN MAP
BITSTRINGS
01000101110...

$$\text{Compression Ratio} = \frac{\text{byte(original image)}}{\text{byte(bitstrings)}}$$


DECOMPRESS


Decode
Indices_to_subband
Ungroup
Dequantize
_recreate
reconstruct
image


BITSTRINGS
01000101110...
HUFFMAN MAP
Indices, extra

Decode

Indices_to_subband

Ungroup

Dequantize

_recreate

reconstruct

image

INDICES

$i_j < 100$: array zeros, size = $i_j + 1$

$i_j = 104$: array zeros, $101 \leq size < 256$

$i_j = 105$: array zeros, $size \geq 256$

$i_j < 100$: $74 < number < 256$

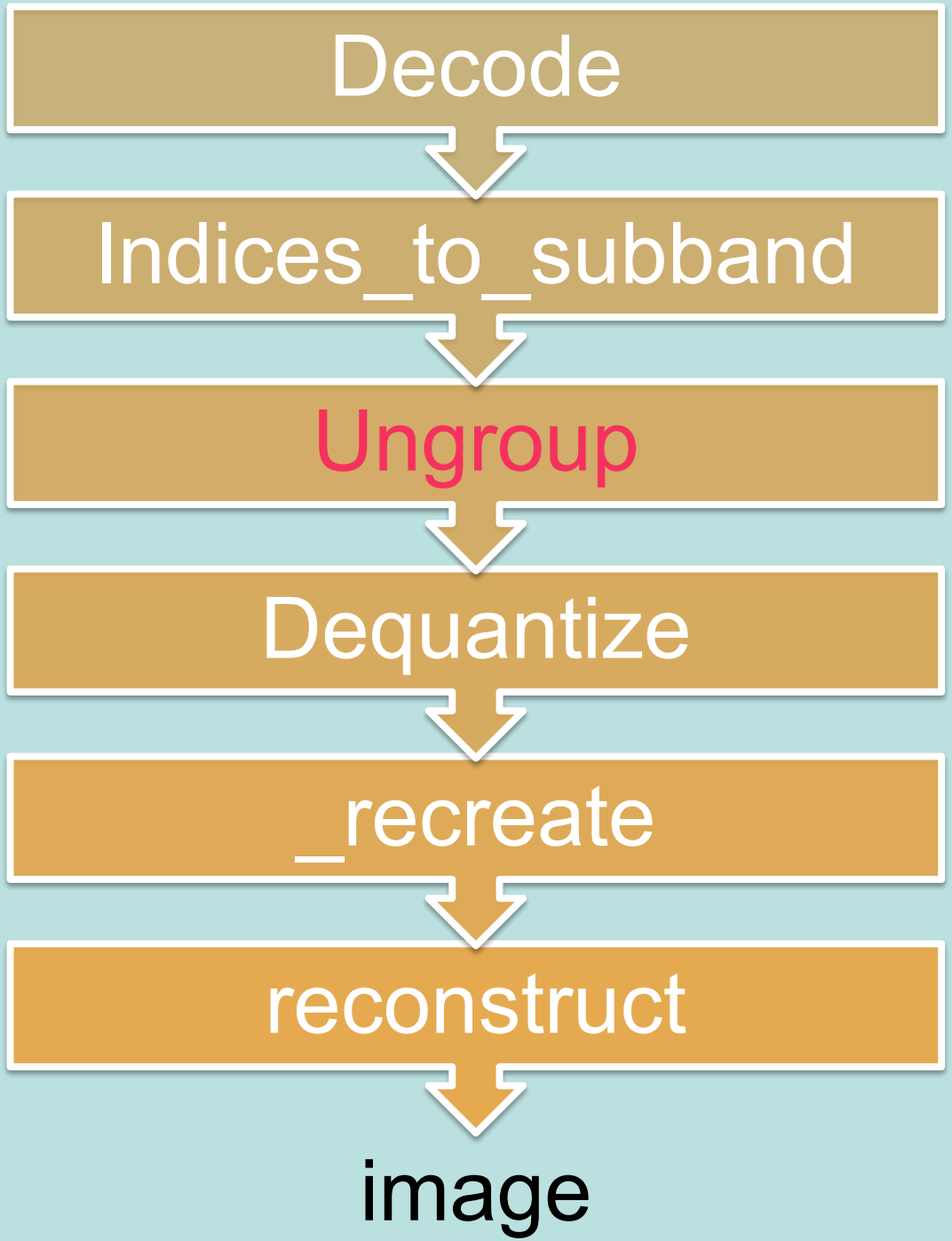
$i_j < 102$: $number \geq 256$

$i_j < 101$: $-73 < number < -256$

$i_j < 103$: $number \leq -256$

Other: number $(i_j - 179)$: $-73 \leq number \leq 74$

GROUP SUBBANDS



GROUP SUBBANDS (GS): 3 groups

↓

TRACK OF SUBBANDS (TS)

i = 0

1. ts = True

Array(gs[j][i: i+ l], shape) $\overset{add}{\rightarrow}$ subbands

i += l

2. ts = False

Array zeros(shape) $\overset{add}{\rightarrow}$ subbands

↓

SUBBANDS

Decode

Indices_to_subband

Ungroup

Dequantize

_recreate

reconstruct

image

SUBBANDS

Cho C = 0.44

$$\hat{a}_k(m,n) = \begin{cases} (p_k(m,n) - C)Q_k + \frac{Z_k}{2}, p_k(m,n) > 0 \\ 0 \qquad\qquad\qquad\qquad , p_k(m,n) = 0 \\ (p_k(m,n) + C)Q_k - \frac{Z_k}{2}, p_k(m,n) > 0 \end{cases}$$

SUBBANDSDQ

Decode

Indices_to_subband

Ungroup

Dequantize

_recreate

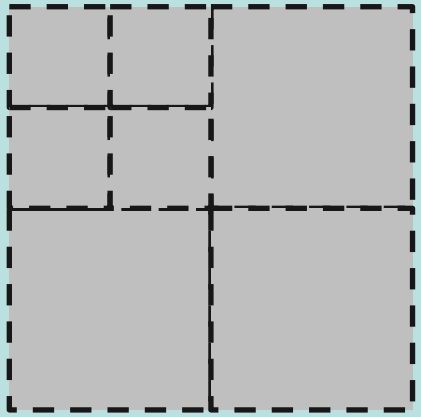
reconstruct

image

SUBBANDSQT

COEFFICIENTS

$LL_n, (LH_n, HL_n, HH_n), \ldots, (LH_1, HL_1, HH_1))$



Decode

Indices_to_subband

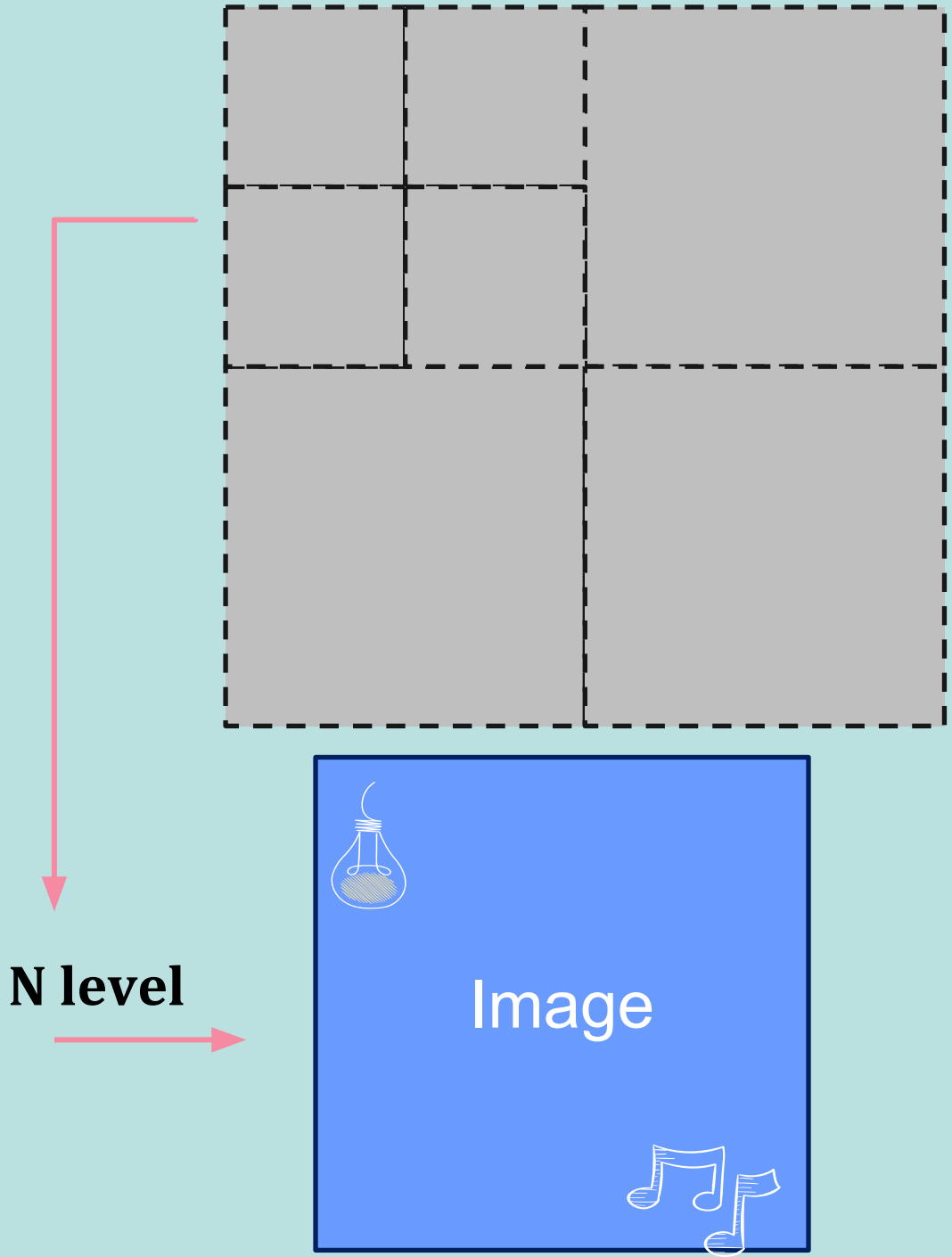
Ungroup

Dequantize

_recreate

reconstruct

image



THANKS FOR YOUR WATCHING