Image coding

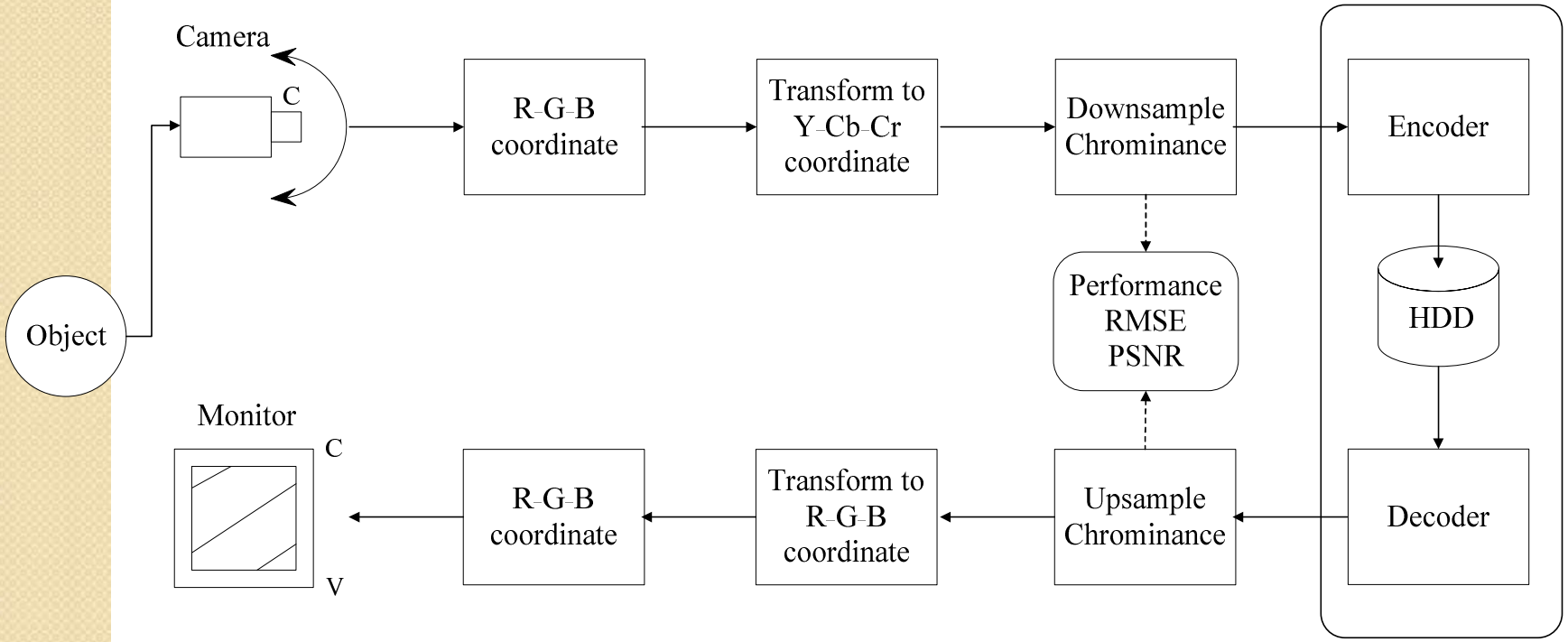
Ary Shiddiqi

ary.shiddiqi@if.its.ac.id

Outline

- Image Compression Fundamentals
- Reduce correlation between pixels
- Quantization and Source Coding
- Overview of Image Compression Algorithms

General Image Storage System



Color Specification

- **Luminance**
  - Received brightness of the light, which is proportional to the total energy in the visible band.
- **Chrominance**
  - Describe the perceived color tone of a light, which depends on the wavelength composition of light
  - Chrominance is in turn characterized by two attributes
    - **Hue**
      - Specify the color tone, which depends on the peak wavelength of the light
    - **Saturation**
      - Describe how pure the color is, which depends on the spread or bandwidth of the light spectrum

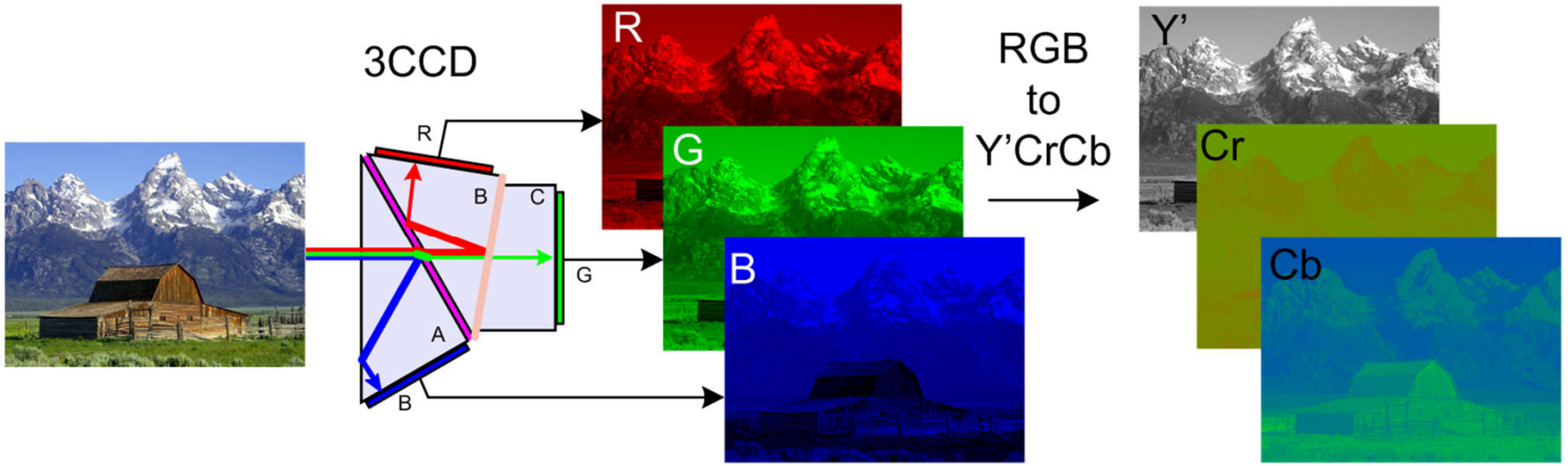
YUV Color Space

- In many applications, it is desirable to describe a color in terms of its luminance and chrominance content separately, **to enable more efficient processing and transmission of color signals**
- One such coordinate is the YUV color space

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.334 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 0 \\ 128 \\ 128 \end{pmatrix}$$

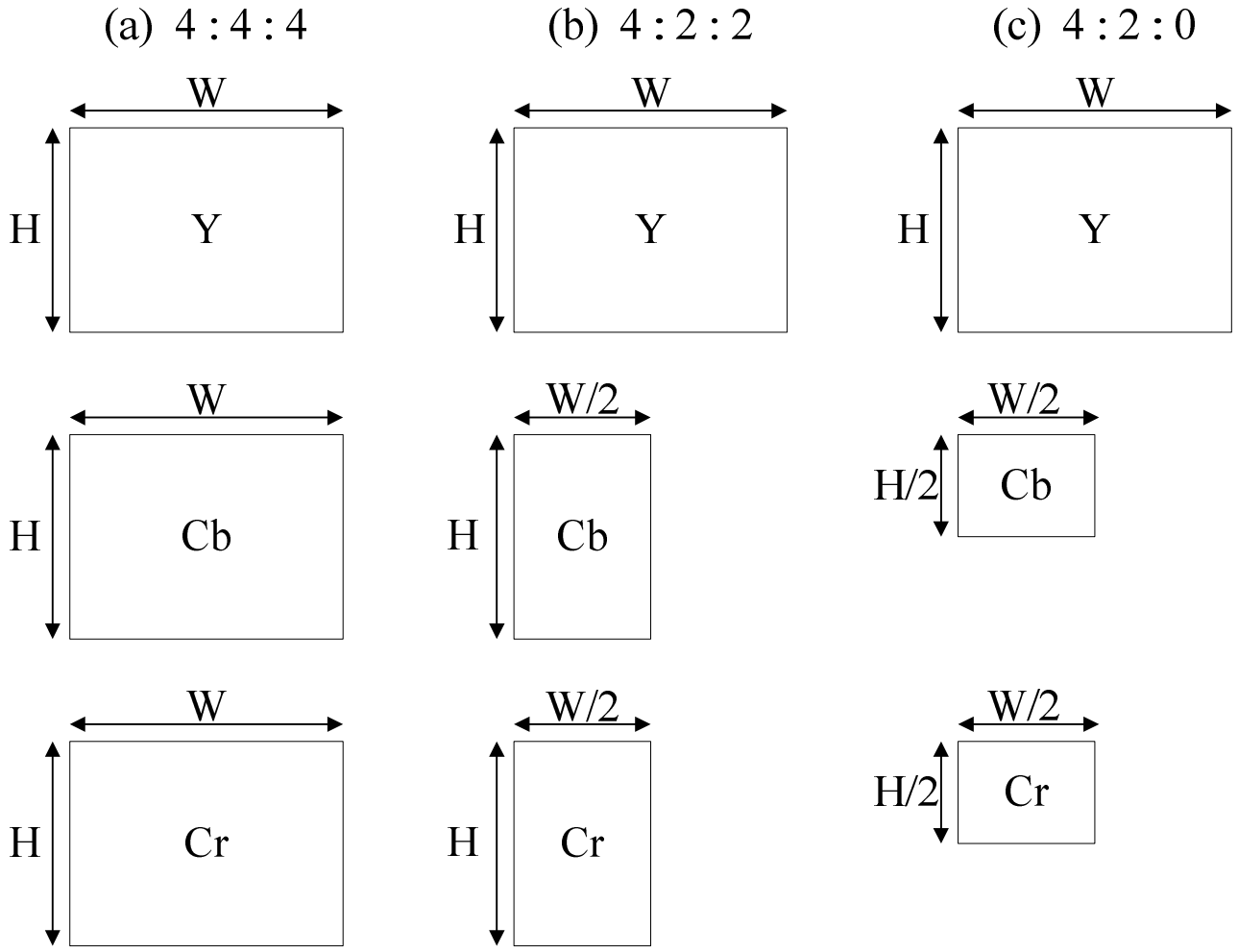
  - Y is the components of luminance
  - Cb and Cr are the components of chrominance
  - The values in the YUV coordinate are related to the values in the RGB coordinate by

Example of RGB to Y'CrCb transformation



Spatial Sampling of Color Component

The three different chrominance downsampling format



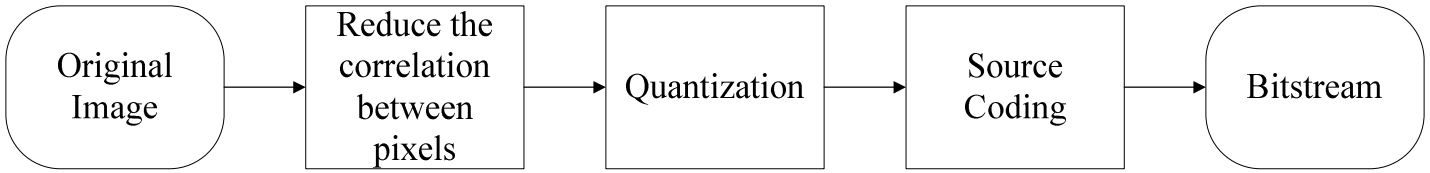
The Flow of Image Compression (1/2)

- What is the so-called image compression coding?
  - To store the image into bit-stream as compact as possible and to display the decoded image in the monitor as exact as possible
- Flow of compression
  - The image file is converted into a series of binary data, which is called the bit-stream
  - The decoder receives the encoded bit-stream and decodes it to reconstruct the image
  - The total data quantity of the bit-stream is less than the total data quantity of the original image

The Flow of Image Compression (2/2)

- Measure to evaluate the performance of image compression
  - Root Mean square error: $RMSE = \sqrt{\frac{\sum_{x=0}^{W-1}\sum_{y=0}^{H-1}[f(x,y)-f'(x,y)]^2}{WH}}$
  - Peak signal to noise ratio: $PSNR = 20\log_{10}\frac{255}{MSE}$
  - Compression Ratio: $Cr = \frac{n1}{n2}$

    Where *n1* is the data rate of original image and *n2* is that of the encoded bit-stream
- The flow of encoding
  - Reduce the correlation between pixels
  - Quantization
  - Source Coding



Outline

- Image Compression Fundamentals
- Reduce correlation between pixels
- Quantization and Source Coding
- Overview of Image Compression Algorithms

Reduce the Correlation between Pixels

- **Orthogonal Transform Coding**
  - KLT (Karhunen-Loeve Transform)
    - Maximal Decorrelation Process
  - DCT (Discrete Cosine Transform)
    - JPEG is a DCT-based image compression standard, which is a lossy coding method and may result in some loss of details and unrecoverable distortion.
- **Subband Coding**
  - DWT (Discrete Wavelet Transform)
    - To divide the spectrum of an image into the lowpass and the highpass components, DWT is a famous example.
    - JPEG 2000 is a 2-dimension DWT based image compression standard.
- **Predictive Coding**
  - DPCM
    - To remove mutual redundancy between seccessive pixels and encode only the new information

The Orthogonal Transform

- **The Linear Transform**
  - The forward transform $y = Ax$
  - The inverse transform $x = Vy$
  - If we want to obtain the inverse transform, we need to compute the inverse of the transform matrix since

$$V = A^{-1}$$

$$VA = A^{-1}A = I$$

$$x = Vy = V(Ax) = (A^{-1}A)x = x$$

- **The Orthogonal Transform**
  - The forward transform $y = V^T x$
  - The inverse transform $x = Vy$
  - If we want to obtain the inverse transform, we need not to compute the inverse of the transform matrix since

$$VV^T = V^T V = I$$

$$x = Vy = V(V^T x) = (V^T V)x = x$$

Karhunen-Loeve Transform

- **KLT is** the optimum transform coder that is defined as the one that minimizes the mean square distortion of the reproduced data for a given number of total bits

The KLT algorithm

X: The input vector with size N-by-1

A: The transform matrix with size N-by-N

Y: The transformed vector with size N-by-1, and each components v(k) are mutually uncorrelated

C*xixj*: The covariance matrix of *xi* and *xj*

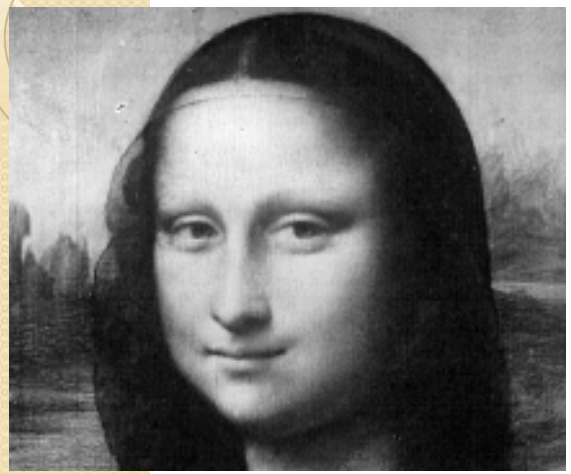
C*yiyj*: The covariance matrix of *yi* and *yj*

The transform matrix **A** is composed of the eigenvectors of the autocorrelation matrix **C***xixj*, which makes the output autocorrelation matrix **C***yiyj* be composed of the eigenvalues in the diagonal direction.

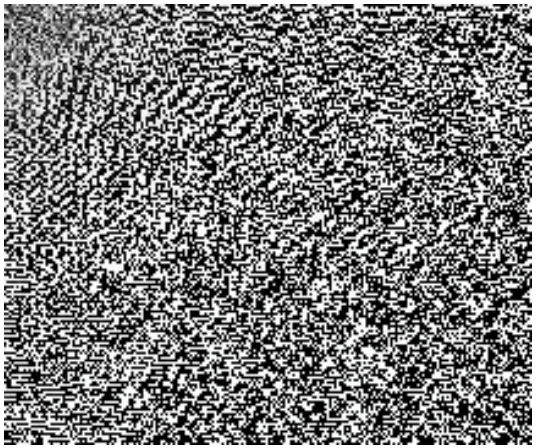
Discrete Cosine Transform (1/2)

- The DCT can concentrate the energy of the transformed signal in low frequency
- This transform has the advantage of concentrating most of the image "energy" or information, in a few frequency components. In other words, energy compaction
- Therefore making a good choice for encoding.

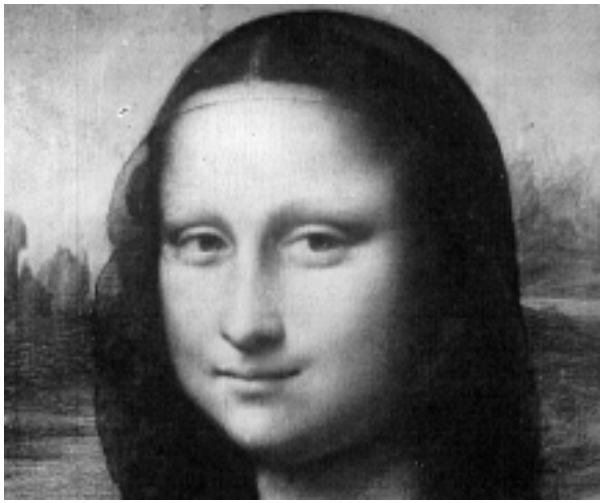
Discrete Cosine Transform (2/2)



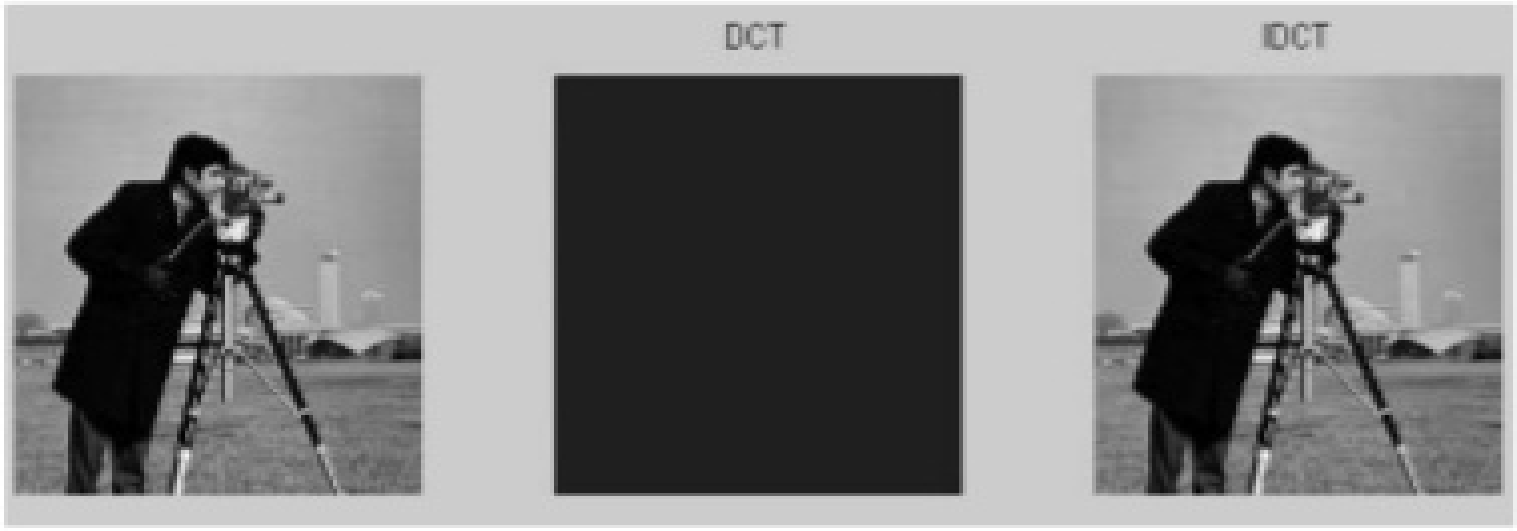
input



transformed



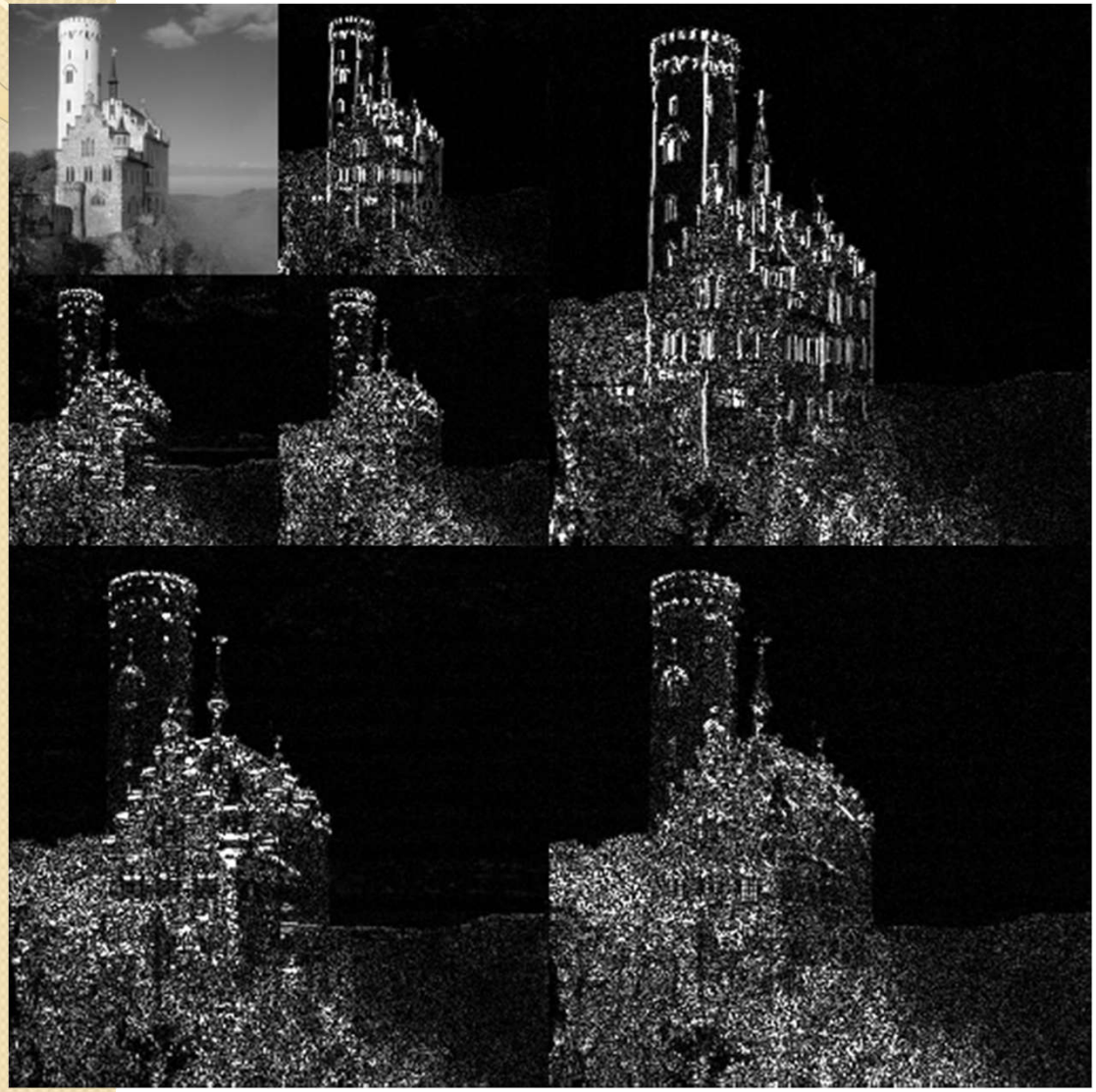
inversed



Discrete Wavelet Transform (1/2)

- Subband Coding
  - The spectrum of the input data is decomposed into a set of bandlimitted components, which is called subbands
  - Ideally, the subbands can be assembled back to reconstruct the original spectrum without any error
- The input signal will be filtered into lowpass and highpass components through analysis filters
- The human perception system has different sensitivity to different frequency band
  - The human eyes are less sensitive to high frequency-band color components
  - The human ears is less sensitive to the low-frequency band less than 20 Hz and high-frequency band larger than 20 KHz

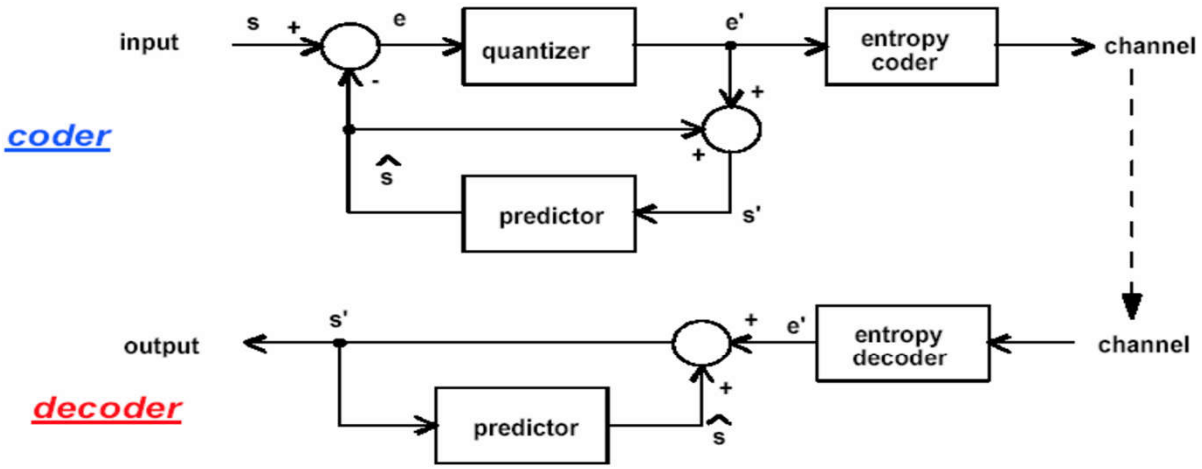
Discrete Wavelet Transform (2/2)



- The original image is high-pass filtered, yielding the three large images, each describing local changes in brightness (details) in the original image.
- It is then low-pass filtered and downscaled, yielding an approximation image; this image is high-pass filtered to produce the three smaller detail images, and low-pass filtered to produce the final approximation image in the upper-left.

Differential pulse-code modulation (DPCM) (1/3)

- uses the baseline of pulse-code modulation (PCM) but adds some functionalities based on the prediction of the samples of the signal.
- The input can be an analog signal or a digital signal.



Prediction error

$$e = s - \hat{s}$$

Reconstruction

$$s' = e' + \hat{s}$$

Reconstruction error = quantization error

$$s' - s = e' - e = q$$

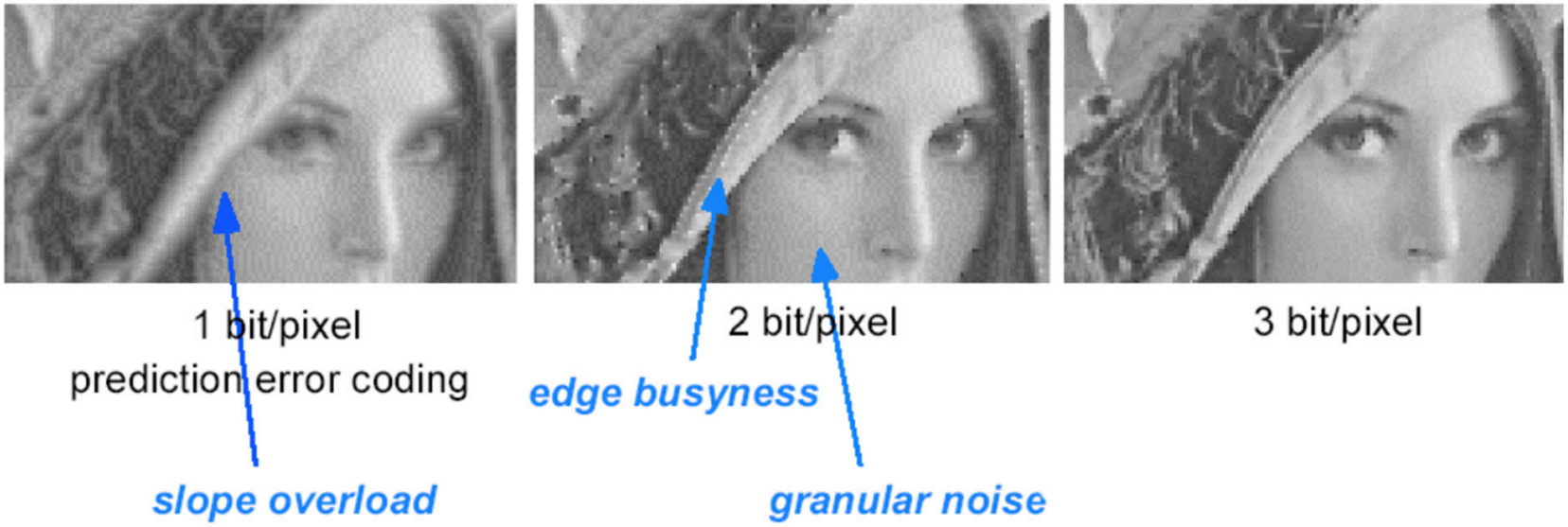
DPCM (2/3)

- Simple example:
  - Code the following value sequence:
  - 1.4 1.75 2.05 2.5 2.4
  - Quantization step: 0.2
  - Predictor: current value = previous quantized value + quantized error.

  - Error = 1.75-1.4 = 0.35 => 0.4
  - Prediction value = 1.4 + 0.4 = 1.8
  - Error = 2.05 - 1.8 = 0.25 => 0.2
  - Prediction value = 1.8 + 0.2 = 2.0
  - Error = 2.5 – 2.0 = 0.5=> 0.4
  - Prediction value = 2.0 + 0.4 = 2.4

  - Send 1.4, 0.4, 0.2, 0.4, …

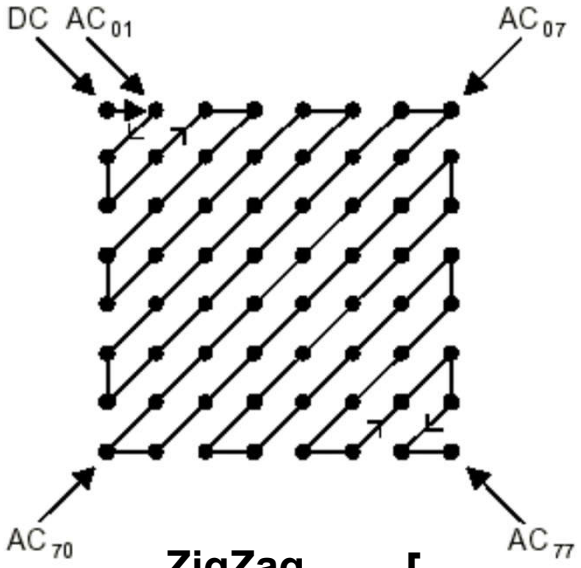
DPCM (3/3)


1 bit/pixel
prediction error coding
2 bit/pixel
3 bit/pixel
edge busyness
slope overload
granular noise

Differential Coding (1/2)

- provide unambiguous signal reception when using some types of modulation.
- It makes data to be transmitted to depend not only on the current signal state (or symbol), but also on the previous one.
- The common types of modulation that require differential coding include phase shift keying and quadrature amplitude modulation.

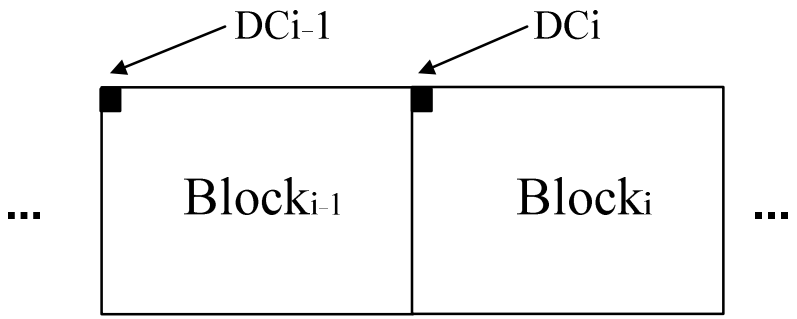
0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63



Differential Coding (2/2)

- We set $DC_0 = 0$.
- DC of the current block DC_i will be equal to $DC_{i-1} + Diff_i$.
- Therefore, the first coefficient is actually the difference of DCs. Then the difference is encoded with Huffman coding algorithm together with the encoding of AC coefficients

Differential Coding : $Diff_i = DC_i - DC_{i-1}$



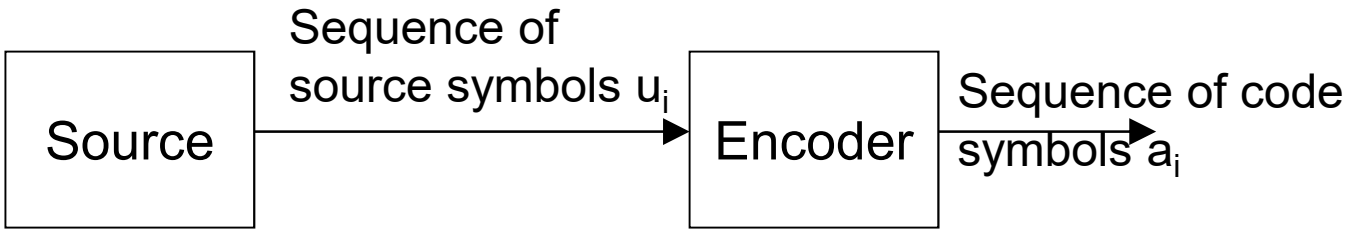
Outline

- Image Compression Fundamentals
- Reduce correlation between pixels
- Quantization and Source Coding
- Overview of Image Compression Algorithms

Quantization and Source Coding

- **Quantization**
  - The objective of quantization is to reduce the precision and to achieve higher compression ratio
  - Lossy operation, which will result in loss of precision and unrecoverable distortion
- **Source Coding**
  - To achieve less average length of bits per pixel of the image.
  - Assigns short descriptions to the more frequent outcomes and long descriptions to the less frequent outcomes
  - Entropy Coding Methods
    - Huffman Coding
    - Arithmetic Coding
    - Run Length Coding
    - Dictionary Codes
      - Lempel-Ziv77
      - Lempel-Ziv 78

Source Coding



Source alphabet

$$U = \{u_1 \quad u_2 \quad \dots \quad u_M\}$$

$$P = \{p_1 \quad p_2 \quad \dots \quad p_M\}$$

Code alphabet

$$A = \{a_1 \quad a_2 \quad \dots \quad a_n\}$$

message

$$u_i \longrightarrow X_1^{(i)}, X_2^{(i)}, \cdots, X_{Ni}^{(i)}$$

where $X_k^{(i)} \in A$, k=1,2,$\cdots$,N_i and i = 1,$\cdots$,M

$X^{(i)}$: codeword

N_i : length of the codeword

Average length of codeword : $\overline{N} = \sum_{i=1}^{M} p(X^{(i)})N_i = \sum_{i=1}^{M} p(u_i)N_i = \sum_{i=1}^{M} p_i N_i$

Huffman Coding (1/2)

- The code construction process has a complexity of $O(N\log_2 N)$
- Huffman codes satisfy the prefix-condition
  - Uniquely decodable: no codeword is a prefix of another codeword

Huffman Coding Algorithm

(1) Order the symbols according to the probabilities

Alphabet set: $S_1, S_2, \ldots, S_N$

Probabilities: $P_1, P_2, \ldots, P_N$

The symbols are arranged so that $P_1 \geqq P_2 \geqq \ldots \geqq P_N$

(2) Apply a contraction process to the two symbols with the smallest probabilities. Replace the last two symbols S_N and S_{N-1} to form a new symbol H_{N-1} that has the probabilities $P_1 + P_2$.

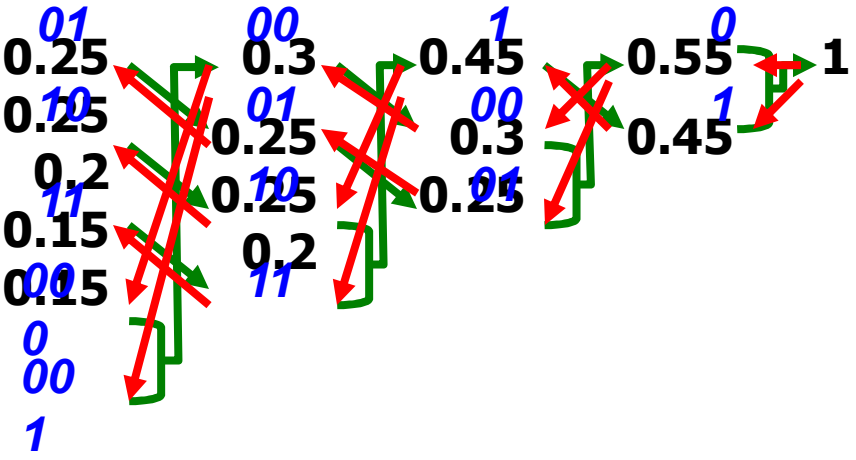
The new set of symbols has N-1 members: $S_1, S_2, \ldots, S_{N-2}, H_{N-1}$

(3) Repeat the step 2 until the final set has only one member.

(4) The codeword for each symbol S_i is obtained by traversing the binary tree from its root to the leaf node corresponding to S_i

Huffman Coding (2/2)

Codeword length	Codeword	X	Probability
2	01	1	0.25
2	10	2	0.25
3	11	3	0.2
3	000	4	0.15
3	001	5	0.15



Arithmetic Coding (1/4)

Shannon-Fano-Elias Coding

- We take **X**={1,2,…,m}, p(x)>0 for all x.
- Modified cumulative distribution function $\overline{F} = \sum_{a<x} P(a) + \frac{1}{2} P(x)$
- Assume we round off $\overline{F(x)}$ to $l(x)$, which is denoted by $\lfloor \overline{F(x)} \rfloor_{l(x)}$
- The codeword of symbol x has *l(x)* $= \left\lceil \log \frac{1}{p(x)} \right\rceil + 1$ bits
- Codeword is the binary value of $\overline{F(x)}$ with *l(x)* bits

x	*P(x)*	*F(x)*	$\overline{F(x)}$	$\overline{F(x)}$ in binary	*l(x)*	codeword
1	0.25	0.25	0.125	0.001	3	001
2	0.25	0.50	0.375	0.011	3	011
3	0.20	0.70	0.600	$0.1\overline{001}$	4	1001
4	0.15	0.85	0.775	$0.1100\overline{011}$	4	1100
5	0.15	1.00	0.925	$0.1110\overline{110}$	4	1110

Arithmetic Coding (2/4)

- **Arithmetic Coding**: a direct extension of Shannon-Fano-Elias coding calculate the probability mass function $p(x^n)$ and the cumulative distribution function $F(x^n)$ for the souece sequence x^n
  - Lossless compression technique
  - Treate multiple symbols as a single data unit

Arithmetic Coding Algorithm

Input symbol is l

$Previous_{low}$ is the lower bound for the old interval

$Previous_{high}$ is the upper bound for the old interval

Range is $Previous_{high}$ - $Previous_{low}$

Let $Previous_{low}$= 0, $Previous_{high}$ = 1, Range = $Previous_{high}$ – $Previous_{low}$ =1

WHILE (input symbol != EOF)

  get input symbol l

  Range = $Previous_{high}$ - $Previous_{low}$

  New $Previous_{low}$ = $Previous_{low}$ + Range* $interval_{low}$ of l

  New $Previous_{high}$ = $Previous_{low}$ + Range* $interval_{high}$ of l

END

Arithmetic Coding (3/4)

Symbol	Probability	Sub-interval
k	0.05	[0.00,0.05)
l	0.2	[0.05,0.25)
u	0.1	[0.20,0.35)
w	0.05	[0.35,0.40)
e	0.3	[0.40,0.70)
r	0.2	[0.70,0.90)
?	0.2	[0.90,1.00)

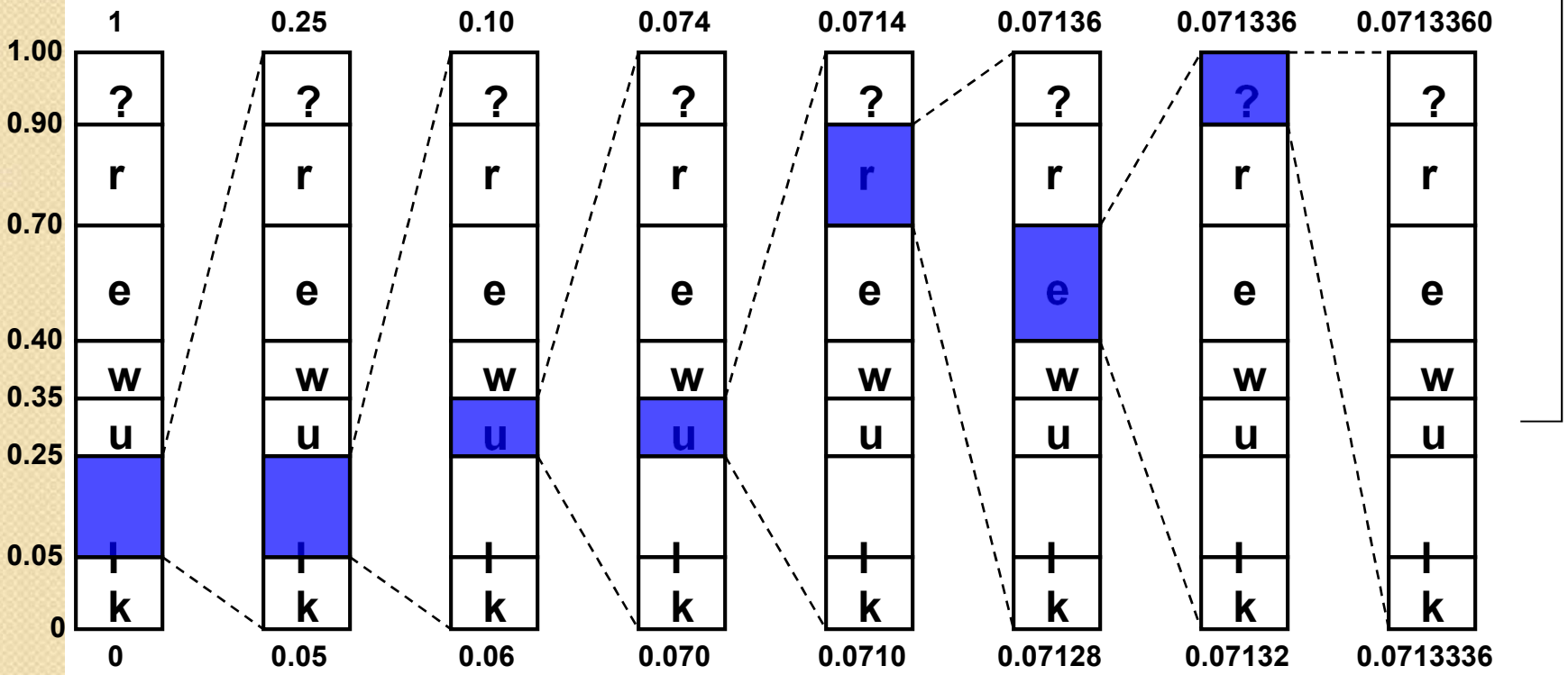
Input String : l l u u r e ?

l l u u r e ?

0.0713348389

$=2^{-4}+2^{-7}+2^{-10}+2^{-15}+2^{-16}$

Codeword : 0001001001000011



Arithmetic Coding (4/4)

Symbol	Probability	Huffman codeword
k	0.05	10101
l	0.2	01
u	0.1	100
w	0.05	10100
e	0.3	11
r	0.2	00
?	0.2	1011

Input String : l l u u r e ?

Huffman Coding **18 bits**
Codeword :
01,01,100,100,00,11,1101

Arithmetic Coding **16 bits**
Codeword : 0001001001000011

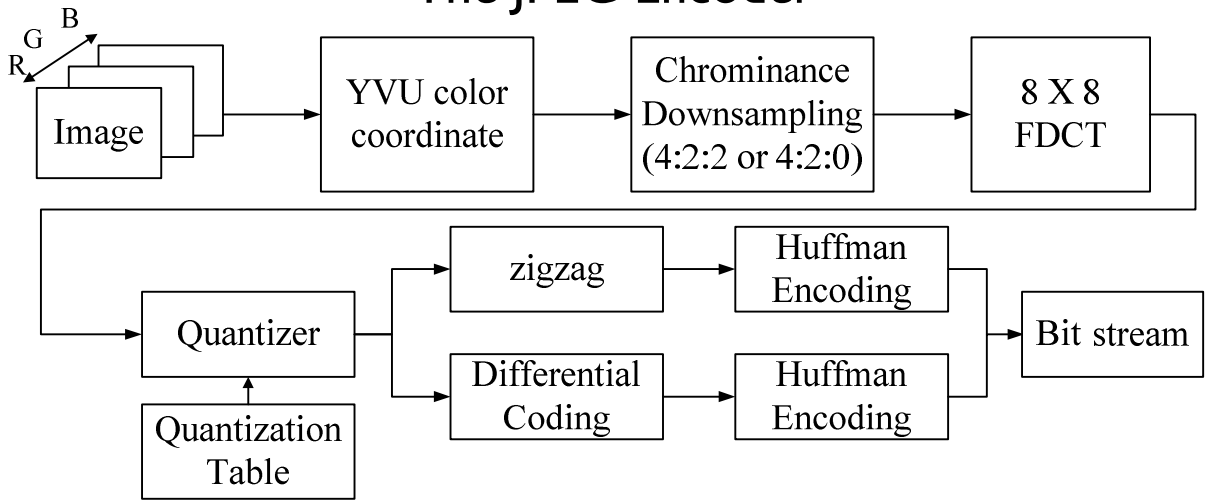
- Arithmetic coding yields better compression because it encodes a message as a whole new symbol instead of separable symbols
- Most of the computations in arithmetic coding use floating-point arithmetic. However, most hardware can only support finite precision
  - While a new symbol is coded, the precision required to present the range grows
  - There is a potential for overflow and underflow
  - If the fractional values are not scaled appropriately, the error of encoding occurs

Outline

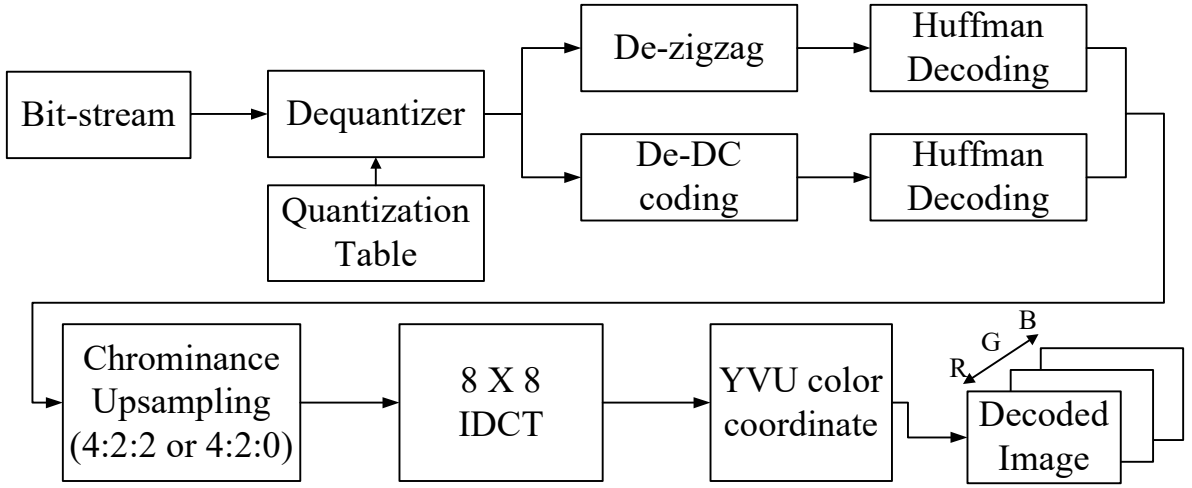
- Image Compression Fundamentals
- Reduce correlation between pixels
- Quantization and Source Coding
- Overview of Image Compression Algorithms

JPEG

The JPEG Encoder

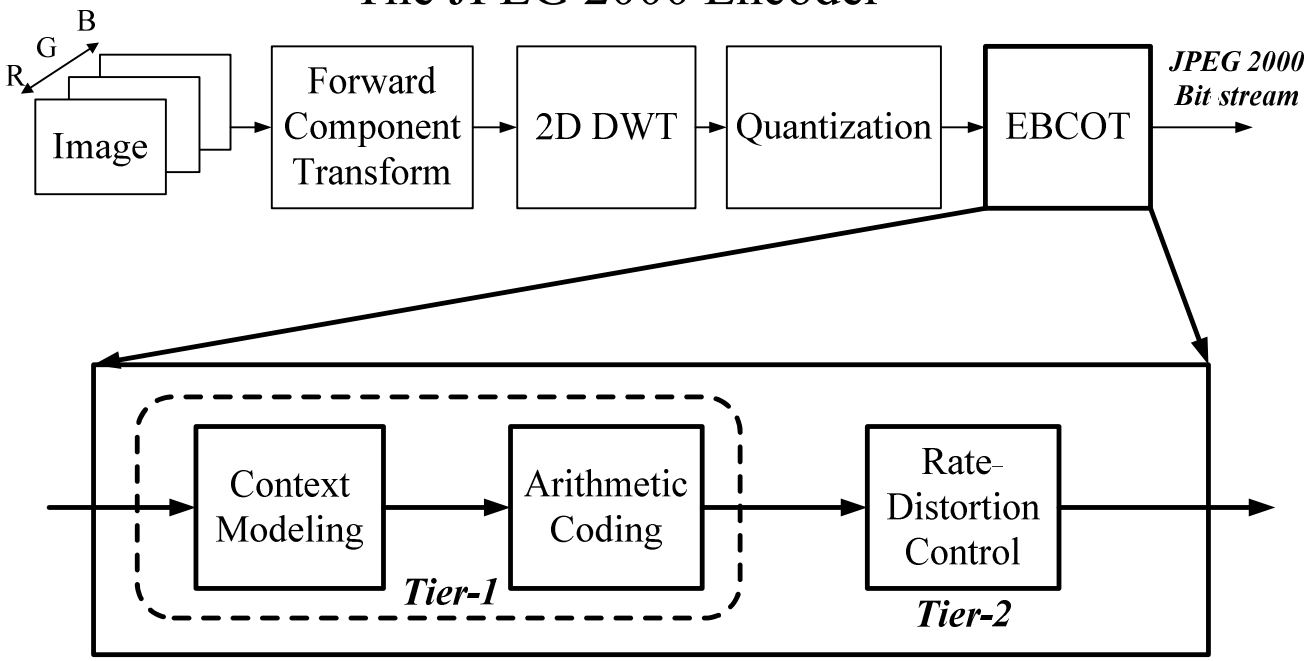


The JPEG Decoder



JPEG 2000

The JPEG 2000 Encoder



The JPEG 2000 Decoder