

# Capítulo 8 – Compressão de Imagens

✱ A compressão de imagens é a arte e a ciência de reduzir o volume de dados necessários para representar uma imagem.

✱ As técnicas de compressão surgiram devido à necessidade de se reduzir o espaço requerido para o armazenamento e o tempo necessário para a transmissão de imagens.

Exemplos:

➤ Imagem colorida com dimensões 1024 x 1024 pixels, cada pixel representado por 24 bits, requer para armazenamento:

$$1024 \times 1024 \times 24 = 25.165.824 \text{ bits} \approx \mathbf{3 \text{ Mbytes}}$$

➤ Vídeo (Filme) → Sequência de quadros de vídeo (frames) onde cada quadro é uma imagem estática colorida.

Suponha um vídeo com resolução 720 x 480 x 24bits (Padrão Standard Definition - SD)

Os reprodutores de vídeo devem exibir os quadros a uma taxa de 30 fps (quadros por segundo – *frames per second*).

Assim, os dados de vídeo digitais SD devem ser acessados em:

$$(30 \text{ frames/s}) \times (720 \times 480 \text{ pixels/frames}) \times (3 \text{ bytes/pixel}) = \mathbf{31.104.000 \text{ bytes/s}}$$

Um filme de duas horas consiste em:

$$31.104.000 \text{ bytes/s} \times (60 \times 60 \text{ s/h}) \times 2 \text{ h} \approx \mathbf{2,24 \times 10^{11} \text{ bytes}} \text{ ou } \mathbf{224 \text{ GB}}$$

2

✱ O termo compressão de dados refere-se ao processo de reduzir o volume de dados necessários para representar dada quantidade de informações

- Dados são os meios pelos quais as informações são transmitidas
- Como várias quantidades de dados podem ser utilizadas para representar a mesma quantidade de informações, dizemos que representações que contêm informações irrelevantes ou repetidas possuem **dados redundantes**.

# Redundância

Seja **R** a redundância relativa de dados da representação com **b** bits

$$\bullet R = 1 - 1/C$$

- onde:  $C = b/b' \rightarrow$  taxa de compressão
  - $b$  e  $b' \rightarrow$  número de bits em duas representações das mesmas informações
- Se  $C = 10$  (taxa de compressão 10:1) significa que a maior representação tem 10 bits de dados para cada 1 bit na representação menor

Redundância (R) é igual a 0.9  $\rightarrow$  indica que 90% de seus dados são redundantes

As redundâncias de dados que podem ser identificadas e exploradas em compressão de imagens são:

- ❖ Redundância de codificação
- ❖ Redundância interpixel
- ❖ Redundância psicovisual

## *Redundância de codificação:*

Um código é um sistema de símbolos (letras, números, bits e assim por diante) utilizados para representar um corpo de informações ou conjunto de eventos. Atribui-se a cada parcela da informação ou evento uma sequência de símbolos de código, denominados palavra de código.

O número de símbolos em cada palavra-código é o seu comprimento ou tamanho.

Os códigos de 8 bits usados para representar as intensidades contêm mais bits do que o necessário para representar estas intensidades.

## *Redundância interpixel*

Como os pixels da maioria dos arranjos de intensidade 2D são correlacionados no espaço (isto é, cada pixel é similar aos pixels vizinhos ou dependente deles), as informações são desnecessariamente replicadas nas representações dos pixels correlacionados.

## *Redundância psicovisual*

A maioria dos arranjos de intensidade 2D contém informações ignoradas pelo sistema visual humano e/ou irrelevantes para a utilização pretendida da imagem. As informações são redundantes no sentido de não serem utilizadas.

# Redundância de codificação

Decorre da representação ineficiente para os valores de pixels, uma vez que é atribuído o mesmo número de bits (ex.  $m$  bits) tanto ao valor de intensidade mais provável quanto ao menos provável.

Presuma que uma variável aleatória discreta  $rk$  no intervalo  $[0, L-1]$  seja utilizada para representar as intensidades de uma imagem  $M \times N$  e que a probabilidade de ocorrência de cada  $rk$  seja  $p_r(rk)$ , dada por:

$$p_r(r_k) = \frac{n_k}{MN} \quad k = 0, 1, 2, \dots, L - 1$$

Se o número de bits utilizados para representar cada valor de  $rk$  for  $l(rk)$ , o número médio de bits necessários para representar cada pixel é:

$$L_{med} = \sum_{k=0}^{L-1} l(r_k) p_r(r_k)$$

Se for utilizado um código de tamanho fixo de  $m$  bits:

$$L_{med} = m$$

O número total de bits necessários para representar uma imagem  $M \times N$  é:

$$MNL_{med}$$

## Exemplo: Ilustração simples da codificação de tamanho variável.

Informações de uma imagem gerada por computador.

$rk$	$pr(rk)$	Código 1	$l1(rk)$	Código 2	$l2(rk)$
87	0.25	01010111	8	01	2
128	0.47	10000000	8	1	1
186	0.25	11000100	8	000	3
255	0.03	11111111	8	001	3
Demais rk	0	-	8	-	0

$L_{med} = 8 \text{ bits}$

$L'_{med}$

$$L'_{med} = 0.25(2) + 0.47(1) + 0.25(3) + 0.03(3) = 1.81 \text{ bits}$$

$$C = (M \times N \times L_{med}) / (M \times N \times L'_{med}) = (256 \times 256 \times 8) / (256 \times 256 \times 1.81) = 4.42$$

$$R = 1 - 1/4.42 = 0.774$$

77,4% dos dados do arranjo de intensidade 2D original de 8 bits são redundantes.

# Redundância Interpixel

Enquanto a redundância de codificação explora a proporção desbalanceada de cada símbolo, a redundância interpixel explora a característica de que pixels vizinhos em uma imagem normalmente possuem alguma relação ou similaridade.

Tipicamente, a maioria dos pixels possui valores não muito diferentes de seus vizinhos, com isso:

- ✓ Valor de cada pixel é previsível a partir dos valores de seus vizinhos;
- ✓ A informação que cada pixel carrega é relativamente pequena;
- ✓ Muito da contribuição visual de um único pixel para uma imagem é redundante, pois ela poderia ser prevista com base nos valores dos pixels adjacentes.

Para aproveitar essa característica, a representação da imagem por uma matriz bidimensional de pixels já não é mais um modelo eficiente, sendo necessário utilizar um formato mais adequado.



A seguir serão citados duas possíveis representações que poderiam ser usadas para reduzir a redundância interpixel.

## 1ª) Armazenar apenas a diferença entre pixels adjacente.

Devido à tendência de pixels vizinhos possuírem valores próximos, essa diferença entre eles geraria, em sua maioria, valores pequenos.

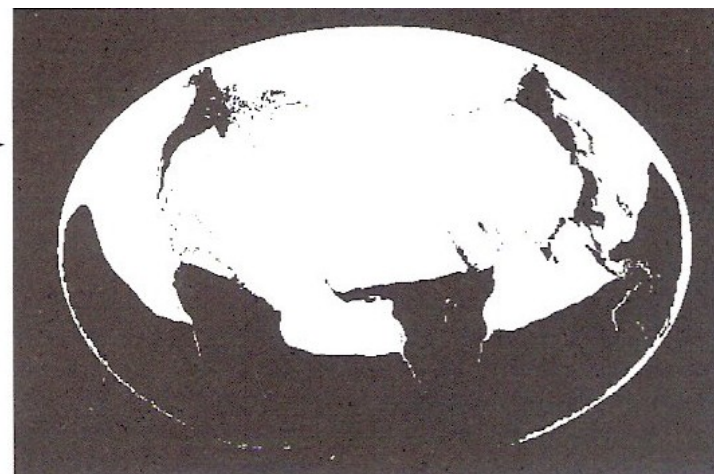
Esse fato aliado a algum método de redução da redundância de codificação, tal como a utilização de códigos de comprimento variável, reduziria ainda mais o número de bits necessários para armazenar uma imagem.

## 2ª) Codificação por comprimento de corrida (*run-length*)

Essa representação explora o fato de que algumas imagens possuem regiões com muitos pixels vizinhos idênticos, como as imagens abaixo:



linha 100 →



280 x 320

Considerando a imagem binária da direita, a codificação por comprimento de corrida percorre cada linha da imagem e, em vez de armazenar o valor preto(0) ou branco(1) para cada pixel, armazena apenas a intensidade ou cor e o número de pixels iguais para cada grupo de pixels idênticos.

As primeiras linhas, que possuem apenas pixels pretos, seriam representadas por (0,320) em que 0 é a cor preta e 320 é o número de pixels com essa cor.

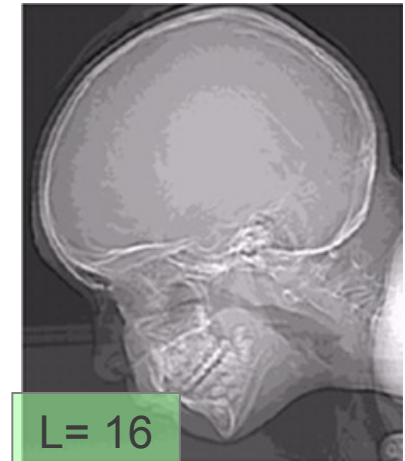
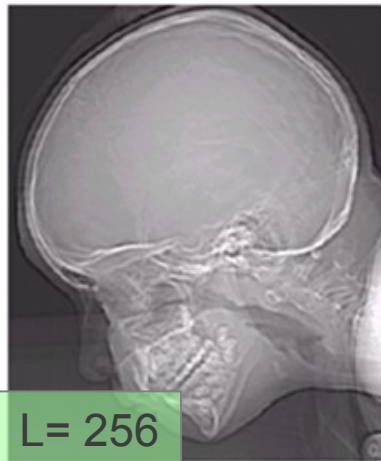
- Capítulo 8 -

# Redundância Psicovisual

A compressão baseada na redundância psicovisual explora a imprecisão do sistema visual humano em perceber certos detalhes em uma imagem.

Essas imprecisões resultam no fato de que o sistema visual humano não responde com a mesma sensibilidade a todas as informações visuais.

As informações que possuem menor importância relativa no processamento visual, denominadas psicovisualmente redundantes, podem ser eliminadas sem prejudicar significativamente a percepção da imagem.



Essa redundância difere das demais por permitir que dados presentes na imagem possam ser eliminados de forma **irreversível**.

Consequentemente, a imagem recuperada não é igual à imagem original.

## Exemplo: Efeito irreversível da redundância psicovisual.

Imagem original com redundância psicovisual.



Histograma

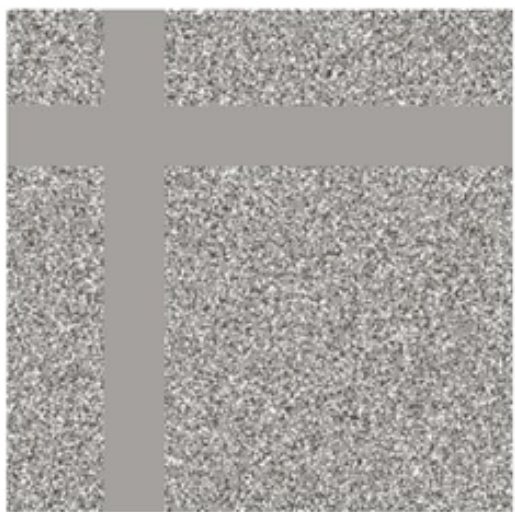
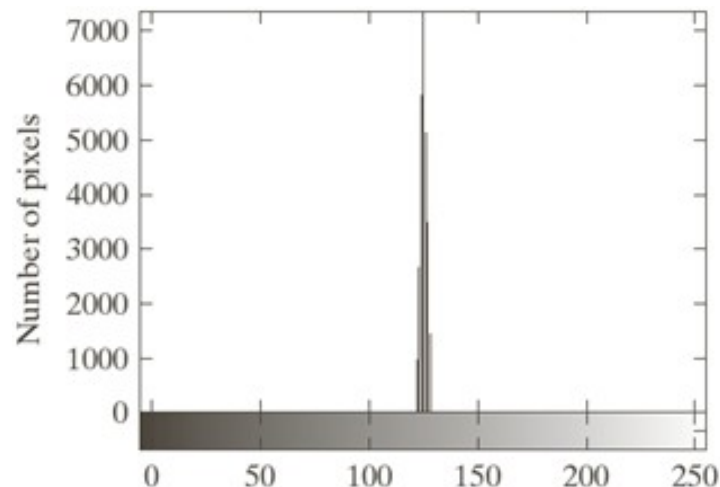


Imagem obtida após um processo de equalização de histograma.

Se a imagem original for representada somente por seu valor médio, a estrutura definida por regiões de intensidade constante e as variações de intensidade aleatórias que a cerca são perdidas.

A decisão de preservar ou não essas informações depende muito da aplicação desejada.

# Medindo as informações da imagem

A teoria da informação possibilita a determinação da quantidade mínima de dados necessária para representar uma imagem sem perda de informação.

A geração da informação pode ser modelada como um processo probabilístico, no qual um evento aleatório  $E$  que ocorre com probabilidade  $P(E)$  contém:

$$I(E) = \log_b \frac{1}{P(E)} = -\log_b P(E) \quad \text{unidades de informação}$$

A quantidade  $I(E)$  é inversamente proporcional a probabilidade de ocorrência do evento  $E$ .

Se  $P(E)=1$ , ou seja, o evento sempre ocorre  $\rightarrow$

$I(E)=0$ , ou seja, nenhuma informação é atribuída a ele.

A base do logaritmo da equação de  $I(E)$  define a unidade utilizada para medir a informação.

Se a base 2 for selecionada, a unidade de informação é um *bit*.

Observe que, se  $P(E)=1/2$  então  $I(E)=-\log_2 1/2$ , isto é, *1 bit*.

Portanto, *1 bit* corresponde à quantidade de informação transferida quando um dos dois eventos possíveis, igualmente prováveis, ocorre.

# Medindo as informações da imagem cont...

Dada uma fonte de eventos aleatórios estatisticamente independentes do conjunto discreto de eventos possíveis  $\{a_1, a_2, \dots, a_J\}$  com probabilidades associadas  $\{P(a_1), P(a_2), \dots, P(a_J)\}$ , a **informação média por saída da fonte**, denominada **entropia** da fonte é dada por:

$$H = -\sum_{j=1}^J P(a_j) \log P(a_j)$$

Se uma imagem for considerada a saída de uma “fonte de intensidades” imaginária, podemos utilizar o histograma da imagem observada para estimar as probabilidades do símbolo da fonte.

Nesse caso, a entropia da fonte de intensidade passa a ser:

$$H = -\sum_{k=0}^{L-1} P_r(r_k) \log_2 P_r(r_k)$$

Como o logaritmo de base 2 é utilizado, a entropia  $H$  é a informação média por saída de intensidade da fonte imaginária em *bits*.

Não é possível codificar os valores de intensidade da imagem de amostra com menos que  $H$  *bits/pixel*.

## Exemplo: Cálculo da entropia de uma imagem.

Seja a imagem 6 x 6 pixels, com cinco níveis de cinza distintos, apresentada abaixo:

93	82	82	77	82	98
77	65	93	98	98	82
82	93	98	77	82	77
82	93	98	77	65	65
93	65	65	98	77	82
77	82	98	65	77	77

A distribuição dos níveis de cinza da imagem é dada pelas probabilidades mostradas na tabela abaixo:

Nível de cinza( $i$ )	0	1	2	3	4
$n_i$	6	9	9	5	7
$p(i)$	0.17	$\frac{0.2}{5}$	0.25	0.14	0.19

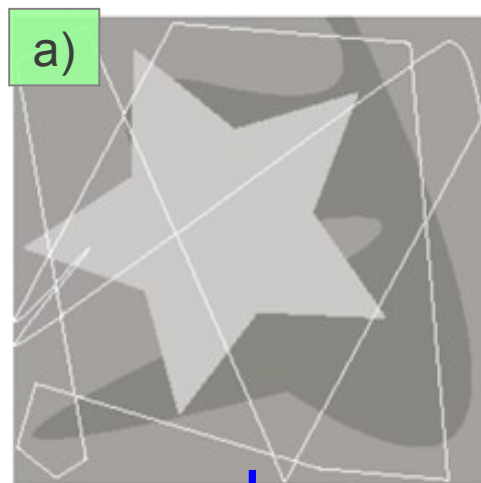
Assim, a entropia desta imagem é dada por:

$$H = - \sum_{i=0}^4 P(i) \log_2 P(i) = - (0.17 \log_2 0.17 + 0.25 \log_2 0.25 + 0.25 \log_2 0.25 + 0.14 \log_2 0.14 + 0.19 \log_2 0.19) \approx 2.29 \text{ bits/pixel}$$

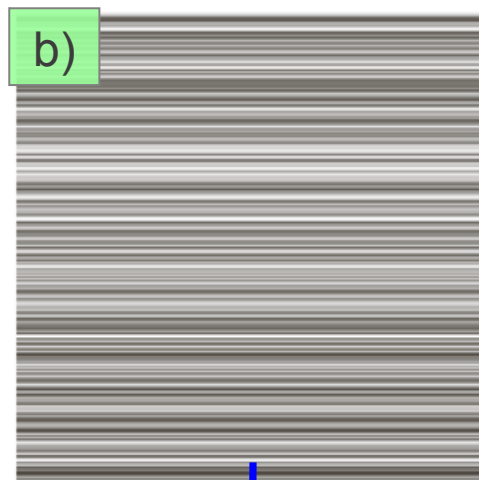


## Exemplo: Comparação da entropia de diferentes imagens.

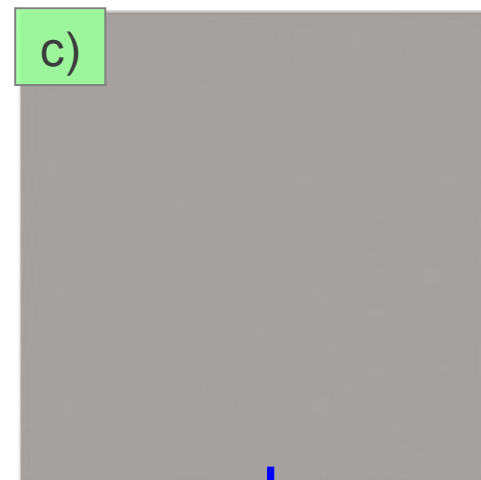
O cálculo da entropia das imagens abaixo resulta em:



$H = 1.6614$  bits/pixel



$H = 8$  bits/pixel



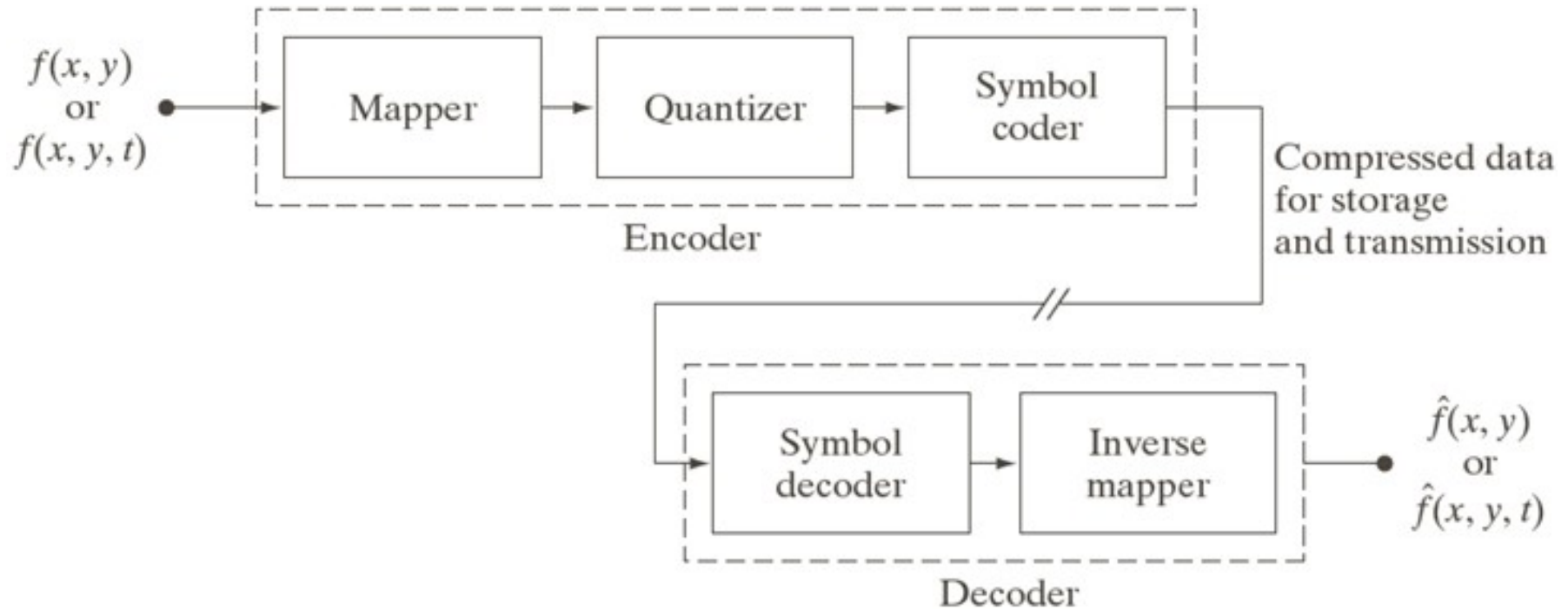
$H = 1.566$  bits/pixel

Observações:

- A imagem a) parece ter mais informações visuais, mas tem quase a entropia calculada mais baixa.
- A imagem b) apresenta quase cinco vezes a entropia da imagem a), mas parece ter aproximadamente as mesmas informações visuais (ou menos).
- A imagem c) que parece ter pouca ou nenhuma informação, apresenta quase a mesma entropia que a imagem a)

A quantidade da entropia da informação em uma imagem não pode ser determinada intuitivamente.

# Sistema Geral de Compressão de Imagens



# Métodos de Compressão de Imagens

## Compressão *sem* perdas

- A imagem, após sua decodificação, deve permanecer exatamente igual à imagem original.
- Exploram principalmente a redundância de codificação e redundância interpixel.
- Imagens médicas, imagens de satélites, documentos digitalizados

Pequena variação na intensidade dos pixels, mesmo que imperceptível ao olho humano, pode afetar o processamento de imagens

## Compressão *com* perdas

- A imagem, após sua decodificação, não permanece exatamente igual à imagem original.
- Exclusão irreversível de parte da informação contida na imagem

# Métodos de Compressão Sem Perda

## 1- Codificação de Huffman

❖ Explora apenas a redundância de codificação.

- ❖ Cada símbolo pode ser substituído por um código de comprimento variável, de forma que os símbolos que ocorrem mais frequentemente na imagem recebem códigos menores que os códigos dos símbolos menos frequentes.
- ❖ Para que cada símbolo seja identificado corretamente no processo de decodificação, nenhum código pode ser prefixo de outro código de comprimento maior.
- ❖ Códigos que satisfazem essa propriedade são conhecidos como *códigos livres de prefixo*.

## Algoritmo:

### 1º Passo:

Realizar um processo chamado de *redução de fontes* → Ordena decrescentemente as probabilidades com que cada símbolo ocorre na imagem e agrupa os dois símbolos com menor probabilidade em um novo símbolo.

Esta operação é repetida até que restem apenas dois símbolos.

A tabela abaixo ilustra o 1º passo da codificação de Huffman:

Original source		Source reduction				
Symbol	Probability	1	2	3	4	
$a_2$	0.4	0.4	0.4	0.4	0.6 0.4	
$a_6$	0.3	0.3	0.3	0.3		
$a_1$	0.1	0.1	0.2	0.3		
$a_4$	0.1	0.1				0.1
$a_3$	0.06	0.1	0.1			
$a_5$	0.04					

# 1- Codificação de Huffman

cont...

## 2º Passo:

Atribuir um código à cada fonte reduzida, iniciando na ordem inversa a que foram obtidas até a fonte original → Como pode ser visto na tabela abaixo, os símbolos binários 0 e 1 são atribuídos arbitrariamente aos dois símbolos da direita. Como o símbolo-fonte reduzido com probabilidade 0.6 é gerado pela combinação de dois símbolos da fonte reduzida à sua esquerda, o 0 utilizado para codificá-lo passa a ser atribuído a esses dois símbolos, e 0 e 1 são arbitrariamente atribuídos a cada um para distingui-los um do outro.

Essa operação é repetida para cada fonte reduzida até a fonte original ser atingida.

Original source		Source reduction							
Symbol	Probability	Code	1	2	3	4	5	6	7
$a_2$	0.4	1	0.4	1	0.4	1	0.4	1	0.6
$a_6$	0.3	00	0.3	00	0.3	00	0.3	00	0.4
$a_1$	0.1	011	0.1	011	0.2	010	0.3	01	
$a_4$	0.1	0100	0.1	0100	0.1	011			
$a_3$	0.06	01010	0.1	0101					
$a_5$	0.04	01011							



Código de Huffman final

❖ O tamanho médio do código de Huffman final é:

$$L_{méd} = (0.4)(1) + (0.3)(2) + (0.1)(3) + (0.1)(4) + (0.06)(5) + (0.04)(5) \\ = 2.2 \text{ bits/pixels}$$

❖ A entropia da fonte é  $2,14$  bits/símbolo.

❖ Depois de o código ser criado, a codificação e/ou decodificação livre de erros é obtida por meio da simples criação de uma tabela de indexação.

❖ Trata-se de um código de blocos instantaneamente decodificável de maneira única.

❖ Qualquer sequência de símbolos codificados pela abordagem de Huffman pode ser decodificada analisando ao símbolos individuais da cadeia da esquerda para a direita.

Ex: Sequência codificada → 0 1 0 1 0 0 1 1 1 1 0 0  
 Primeira palavra-código válida: 0 1 0 1 0 → a3  
 Próximo código válido: 0 1 1 → a1  
 Mensagem completamente decodificada → a3 a1 a2 a2 a6

- ❖ Quando um número grande de símbolos deve ser codificado, a construção de um código de Huffman otimizado torna-se uma tarefa complexa do ponto de vista computacional.
- ❖ Quando probabilidades de símbolo-fonte podem ser antecipadamente estimadas, uma codificação “quase ótima” pode ser atingida com códigos de Huffman **pré-calculados**.
- ❖ Vários padrões populares de compressão de imagens, incluindo os padrões **JPEG** e **MPEG**, especificam tabelas de codificação predeterminadas de Huffman que foram pré-calculadas com base em dados experimentais



## 2- Codificação de Shannon-Fano

É uma técnica bastante simples para geração de códigos livres de prefixo.

### Algoritmo:

1) Os símbolos da fonte são listados em ordem decrescente de probabilidade;

3) Os símbolos são divididos em dois grupos, de forma que as somas das probabilidades nos dois grupos sejam as mais próximas possíveis;

5) Um grupo recebe o valor 1 e o outro recebe o valor 0;

7) O procedimento é repetido para os demais grupos até que reste apenas um símbolo em cada grupo;

9) O código para cada símbolo original da fonte é formado pela sequência resultante de valores 0 e 1.

## 2- Codificação de Shannon-Fano

cont...

A tabela abaixo ilustra um exemplo da aplicação da técnica de Shannon-Fano:

<b>Símbolo</b>	<b>Probabilidade</b>	<b>Código</b>		
S2	0.30	0	0	
S6	0.20	0	1	
S1	0.20	1	0	0
S4	0.10	1	0	1
S3	0.10	1	1	0
S5	0.10	1	1	1

### 3 - Codificação Aritmética

❖ Na codificação aritmética não há correspondência individual entre os símbolos e os códigos.

❖ Conjuntos inteiros de símbolos são codificados em um único intervalo  $[0,1]$ .

❖ O processo de codificação baseia-se em subdivisões sucessivas de intervalos numéricos, proporcionais às probabilidades de cada símbolo.

#### Exemplo:

Suponha que uma sequência ou mensagem de cinco símbolos  $a_1 a_2 a_3 a_3 a_4$ , a partir de uma fonte de quatro símbolos, é codificada.

No início do processo de codificação, considera-se que a mensagem ocupe todo o intervalo  $[0,1]$  que é subdividido em quatro regiões com base nas probabilidades de cada símbolo-fonte, conforme a tabela abaixo:

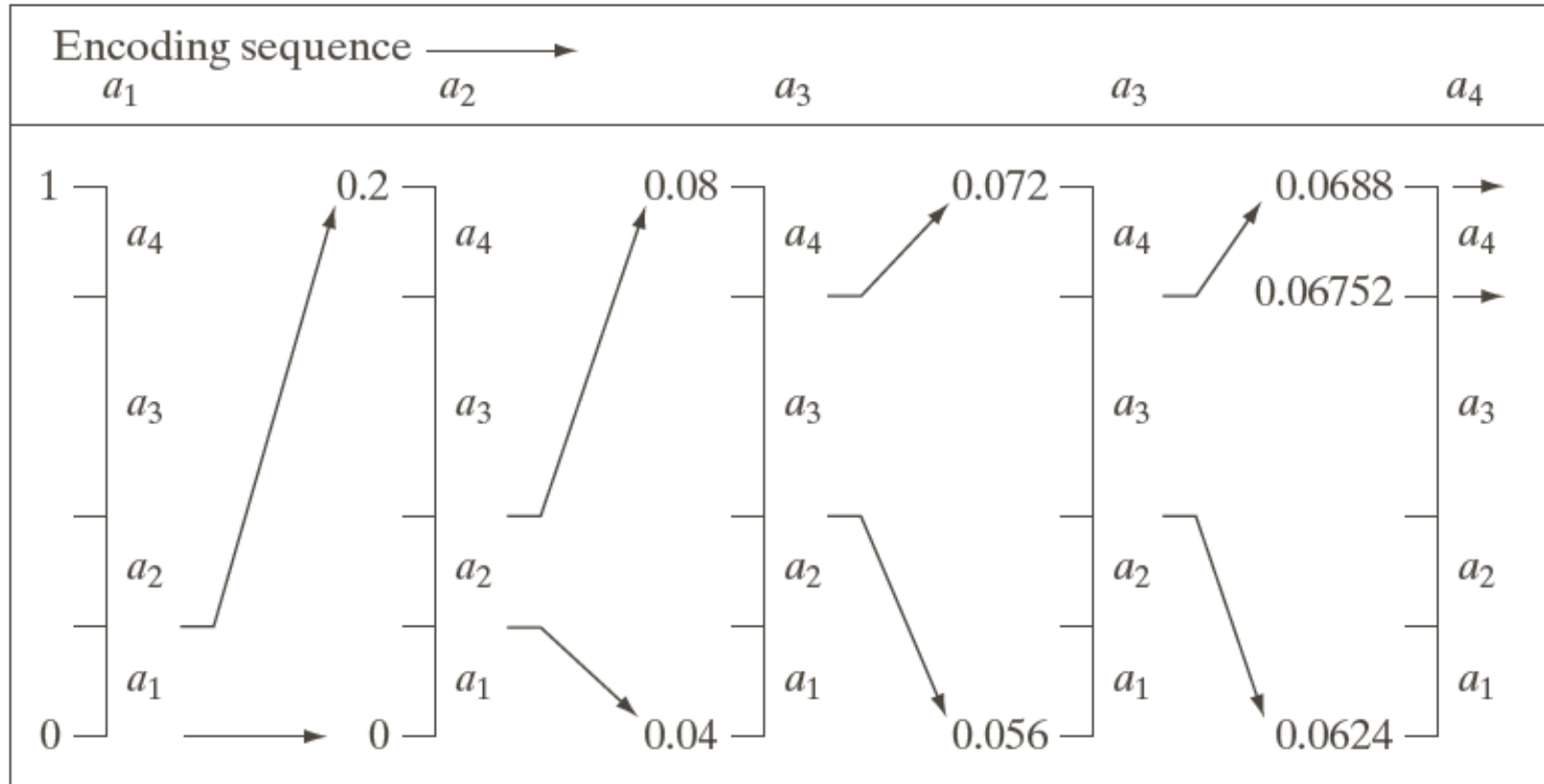
Source Symbol	Probability	Initial Subinterval
$a_1$	0.2	$[0.0, 0.2)$
$a_2$	0.2	$[0.2, 0.4)$
$a_3$	0.4	$[0.4, 0.8)$
$a_4$	0.2	$[0.8, 1.0)$

### 3 - Codificação Aritmética

cont...

O símbolo  $a_1$ , por exemplo, é associado ao subintervalo  $[0,0.2]$ .

Por ser o primeiro símbolo da imagem sendo codificada, o intervalo de mensagem é inicialmente estreitado para  $[0,0.2]$ .



Na mensagem aritmeticamente codificada, três dígitos decimais são utilizados para representar a mensagem de cinco símbolos.

## 4- Codificação de Lempel-Ziv-Welch (LZW)

- ❖ O nome do algoritmo é derivado dos nomes dos desenvolvedores: Abraham Lempel, Jakob Ziv e Terry Welch.
- ❖ É uma codificação baseada em **dicionário** → não requer conhecimento antecipado da probabilidade de ocorrência dos símbolos que serão codificados.
- ❖ É utilizada nos formatos **GIF**, **TIFF** e **PDF**.
- ❖ Explora a redundância interpixel.

## 4- Codificação de Lempel-Ziv-Welch (LZW) ...cont

❖ No início do processo de codificação, é construído um banco de códigos ou **dicionário** contendo os símbolos-fonte a serem codificados.

Ex: Para imagens monocromáticas de 8 bits → As 256 primeiras palavras do dicionário são atribuídas às intensidades 0, 1, 2, ...255.

❖ À medida que o codificador analisa sequencialmente os pixels da imagem, as sequências de intensidade que não estão contidas no dicionário são adicionadas à próxima posição livre do dicionário.

Ex: Se os dois primeiro pixels da imagem forem brancos →

A sequência “255 – 255” pode ser atribuída à posição 256 do dicionário.

Da próxima vez que dois pixels brancos consecutivos forem encontrados, a palavra-código na posição 256 do dicionário é utilizada para representá-los.

Supondo dicionário de 9 bits (512 palavras):

Os (8 + 8) bits originais são substituídos por uma palavra código de 9 bits

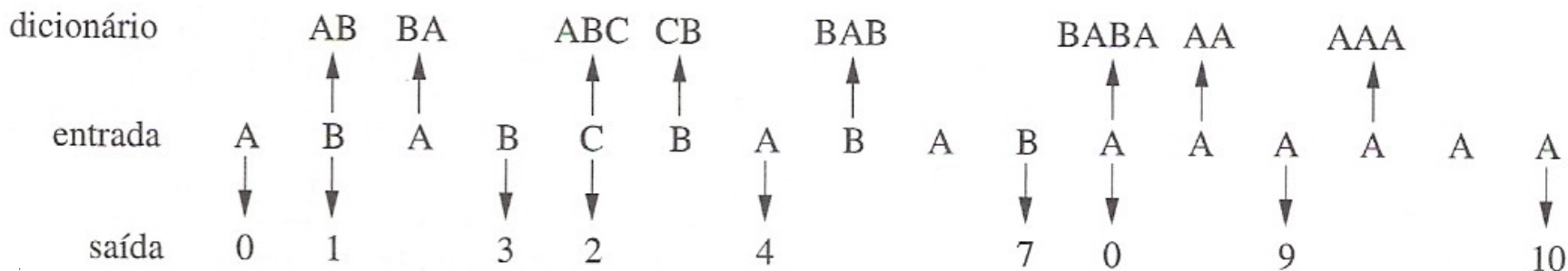
# 4- Codificação de Lempel-Ziv-Welch (LZW) ...cont

- ❖ O tamanho do dicionário é um importante parâmetro do sistema:
  - Se for pequeno demais, a detecção de sequências de nível de intensidade correspondente será menos provável;
  - Se for grande demais, o tamanho das palavras-código afetará desfavoravelmente o desempenho da compressão.

**Exemplo 1:** Seja uma fonte capaz e gerar símbolos {A,B,C}

Sequência de entrada: A B A B C B A B A B A A A A A

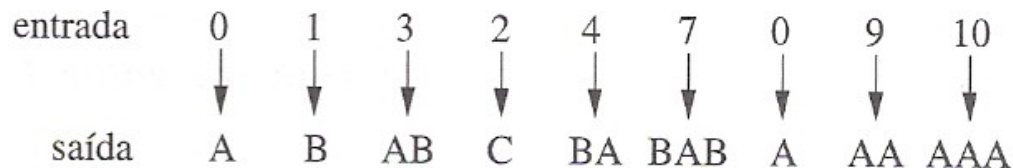
CODIFICAÇÃO



## DICIONÁRIO

Índice	0	1	2	3	4	5	6	7	8	9	10
Entrada	A	B	C	AB	BA	ABC	CB	BAB	BABA	AA	AAA

## DECODIFICAÇÃO



## Exemplo 2: Codificação LZW

Considere a imagem e uma borda vertical de 8 bits e dimensões 4 x 4:

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Dicionário de 512 palavras com o seguinte conteúdo inicial:

Posição no dicionário	Entrada
0	0
1	1
...	...
255	255
256	-
...	...
511	-

Currently Recognized Sequence	Pixel Being Processed	Encoded Output	Dictionary Location (Code Word)	Dictionary Entry
	39			
39	39	39	256	39-39
39	126	39	257	39-126
126	126	126	258	126-126
126	39	126	259	126-39
39	39			
39-39	126	256	260	39-39-126
126	126			
126-126	39	258	261	126-126-39
39	39			
39-39	126			
39-39-126	126	260	262	39-39-126-126
126	39			
126-39	39	259	263	126-39-39
39	126			
39-126	126	257	264	39-126-126
126		126		

Na conclusão da codificação o dicionário tem 265 palavras-código.

Redução da imagem original de 128 bits para 90 bits (dez códigos de 9 bits).

Taxa de compressão resultante: 1,42 : 1



# 5- Codificação *Run-Length* (RLE)

## ❖ Em português: Codificação por Comprimento de Corrida

❖ Explora a redundância interpixel de modo a armazenar, para cada sequência de pixels iguais, apenas seu valor e o número de ocorrências.

Ex: Codificação RLE de uma linha → **(20,29) (35,9) (170,55) (220,4) (15,8) (20,3)**

Intensidade      No de ocorrências

❖ É utilizada nos formatos CCITT, JBIG2, JPEG, M-JPEG, M-PEG, BMP.

❖ Quando há poucas (ou nenhuma) sequências de pixels idênticos, a codificação RLE resulta na expansão dos dados.

❖ Mais apropriada para imagens cujos símbolos se repetem com grande frequência, por exemplo, em imagens binárias.

❖ Para imagens binárias:

- Por haver apenas duas intensidades possíveis (preto e branco), os pixels adjacentes têm mais chance de serem idênticos.
- Cada linha da imagem pode ser representada apenas por uma sequência de tamanhos – em vez de pares de tamanhos e intensidades.

# 5- Codificação *Run-Length* (RLE)

...cont

❖ Duas abordagens de codificação RLE para imagens binárias:

<sup>1º</sup> Requer as posições iniciais e os comprimentos das corridas com valor 1;

<sup>2º</sup> Utiliza apenas os comprimentos das corridas, iniciando-se com o comprimento das corridas de valor 1.

**Exemplo 1:** Codificação RLE de três linhas de uma imagem binária.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1	1	1	1	0	0	0	1	1	0	0	0	1	1	1	1	0	1	1	0	1	1	1
2	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
3	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

**1ª Abordagem:**

Linha 1 → (1,3),(7,2),(12,4),(17,2),(20,3)

Linha 2 → (5,13),(19,4)

Linha 3 → (1,3),(17,6)

**2ª Abordagem:**

Linha 1 → 3, 3, 2, 3, 4, 1, 2, 1, 3

Linha 2 → 0, 4, 13, 1, 4

Linha 3 → 3, 13, 6

## 5- Codificação *Run-Length* (RLE)

...cont

**Exemplo 2:** Codificação RLE de uma imagem monocromática não binária.

- Supondo que cada nível da imagem digital esteja codificado com 1 byte (8bits), o dado da imagem original X, abaixo, está representado com 28 bytes.

• Imagem original



X=

15	12	12	10	12	12	15
12	12	10	10	10	12	12
12	10	10	10	10	10	12
10	10	10	10	10	10	10

**28 bytes**

• Codificação RLE



Y=

01	15	02	12	01	10	02	12	01	15
02	12	03	10	03	12				
05	10	01	12						
07	10								

**22 bytes**

# Formato de Arquivo BMP

❖ O formato BMP foi projetado para sistemas operacionais que rodem sobre a plataforma INTEL (Windows e OS/2).

❖ Arquivos BMP são armazenados no formato DIB (**Device-Independent Bitmap**) que permite exibir a imagem em qualquer dispositivo; ou seja, o bitmap especifica a cor do pixel numa forma independente do método usado pelo dispositivo para representá-la.

❖ A extensão padrão dos arquivos DIB do Windows é ".BMP".

❖ Versões de BMP quanto a quantidade de cores:

- 1 bit/pixel (2<sup>1</sup> = 2 cores)
- 4 bits/pixel (2<sup>4</sup> = 16 cores)
- 8 bits/pixel (2<sup>8</sup> = 256 cores)
- 24 bits/pixel (true color com até 2<sup>24</sup> ≈ 16 milhões de cores)
- 32 bits/pixels (true color com até 2<sup>32</sup> ≈ 4 bilhões de cores)

❖ É muito raro, mas arquivos de formato BMP podem, nas versões de **4 e 8 bits/pixel**, utilizar a compressão RLE (*Run-length encoding*), de forma a reduzir o tamanho do arquivo que armazena o Bitmap.

❖ Estrutura geral do formato BMP:

a) Cabeçalho de arquivo → Informações do arquivo

Contém a assinatura BM e informações sobre o tamanho e lay-out do arquivo BMP (disposição dos dados dentro do arquivo).

c) Cabeçalho de mapa de bits → Informações da imagem

Contém as informações da imagem contida no arquivo. Define as dimensões, tipo de compressão (se houver) e informações sobre as cores da imagem.

e) Paleta ou mapa de cores → Opcional

Somente estará presente em arquivos de imagens que usam 16 ou 256 cores

(4 e 8 bits/pixel). Nas demais, em seu lugar, vem diretamente a parte seguinte: área de dados da imagem.

g) Área de dados da imagem contida no arquivo

Dados que permitem a exibição da imagem propriamente dita, os dados dos pixels a serem exibidos. Podem ser com ou sem compressão.

# Formato de Arquivo BMP

...cont

**Exemplo:** Exemplo típico de imagem BMP de 16 cores (4 bits/pixel)

## Cabeçalho do arquivo

## Paleta de Cores

Type	BM	<i>Azul</i>	<i>Verde</i>	<i>Vermelho</i>	<i>Indice em hexa</i>
Size	3118	84	252	84	0000000
Reser1	0	252	252	84	0000001
Reser2	0	84	84	252	0000002
OffSetBits	118 (14+54+(4 x 16))	252	84	252	0000003
<b>Cabeçalho da Imagem</b>		84	252	252	0000004
Size	40	252	252	252	0000005
Width	80	0	0	0	0000006
Height	75	168	0	0	0000007
Planos	1	0	168	0	0000008
BitCount	4 (bits/pixel)	168	168	0	0000009
BiCompress	0 (s/compressão)	0	0	168	000000A
TamanhoImagem	3000 (3118 – 118)	168	0	168	000000B
XporMeter	0	0	168	168	000000C
YporMeter	0	168	168	168	000000D
Coresusadas	16	84	84	84	000000E
CoresImportantes	16	252	84	84	000000F

## Área de dados da imagem

cores de cada pixel da imagem....

.....

3000 Bytes com dados da imagem....

.....

## Compressão RLE de imagens BMP de 8 bits/pixel

- ✓ Neste caso o campo *BiCompress* do cabeçalho de mapa de bits está setado com *BI\_RLE8 (01)*.
- ✓ Neste formato o RLE tem 2 modos: ***Encoded mode*** e ***Absolute mode***. Estes dois modos podem coexistir na imagem.

### Encoded mode

Este modo usa compressão RLE de 2 bytes.

O 1º byte especifica o número de pixels consecutivos que serão desenhados usando o índice de cor contido no 2º byte.

## Absolute mode

Este modo é sinalizado pelo 1º byte do par ser setado com ZERO.

O 2º byte indica uma de quatro condições possíveis, mostradas na tabela abaixo:

Valor do 2º byte	Condição
0	Fim da linha
1	Fim da imagem
2	Delta → é entendido como um deslocamento do próximo pixel a ser representado na tela. Os 2 bytes seguintes (ao flag 2) contêm valores que indicam o deslocamento horizontal e vertical do próximo pixel a partir da posição corrente.
3 - 255	O valor do 2º byte representa o número de bytes seguintes que serão descritos na forma não comprimida isso é: cada qual com seu valor representando um índice de cor da paleta.



# Formato de Arquivo BMP

...cont

**Exemplo:** Exemplo de uso dos 2 modos de compressão RLE

Suponha que no arquivo, na área de dados da imagem, estejam os valores:

03 04 05 06 00 03 45 56 67 02 78 00 02 05 01 02 78 00 00 09 1E 00 01

Acompanhe na tabela abaixo o que seria representado na matriz de dados da imagem, quando os dados fossem expandidos:

## •Dados

03 04           s  
05 06  
00 03 45 56 67  
02 78  
00 02 05 01  
02 78  
00 00  
09 1E  
00 01

## •Dados expandidos

04 04 04  
06 06 06 06 06  
45 56 67  
78 78  
delta de 5 pixels p/esquerda e 1 p/baixo  
78 78  
Fim de linha  
1E 1E 1E 1E 1E 1E 1E 1E 1E  
Fim da Imagem

# Métodos de Compressão com Perda

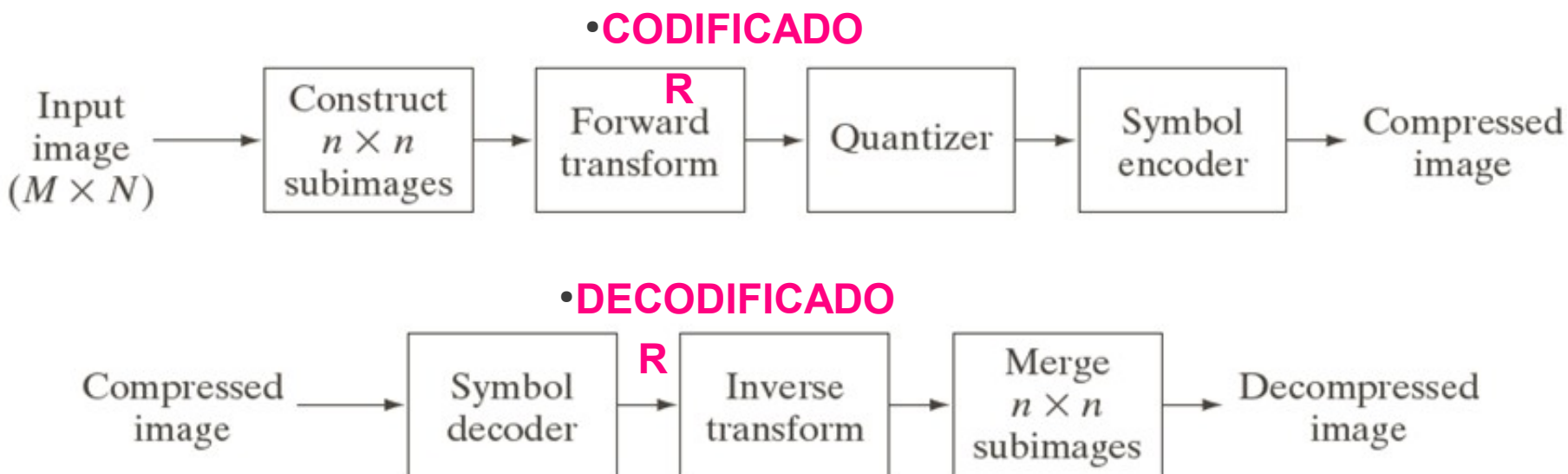
- ❖ Utilizada quando o tamanho da imagem é mais importante que a qualidade.
- ❖ Detalhes que a visão humana não percebe, ou percebe apenas com dificuldade.
- ❖ Não há um limite previamente estabelecido para a taxa de compressão, pois quanto mais detalhes são excluídos ou aproximados durante a codificação, maior será a compressão.
- ❖ Taxa de perda é um parâmetro da compressão:
- ❖ quanto maior a perda admitida, maior compressão se consegue

# 1- Codificação por transformada em blocos

- ❖ Consiste em uma técnica de compressão que divide uma imagem em pequenos blocos não sobrepostos de mesmo tamanho (por exemplo, 8x8) e processa os blocos independentemente utilizando uma transformada 2D.
- ❖ Uma transformada linear e reversível é utilizada para mapear cada *bloco* ou *sub-imagem* em um conjunto de coeficientes da transformada.
- ❖ Os coeficientes são, então, **quantizados** e **codificados**.
- ❖ Para a maioria das imagens, um número significativo dos coeficientes tem pequenas magnitudes e pode ser grosseiramente quantizado (ou, totalmente descartado) com pouca distorção de imagem.
- ❖ Diversas transformadas são comumente utilizadas em compressão de imagens, por exemplo, a transformada discreta do cosseno, a transformada discreta de Fourier, a transformada de Karhunen-Loève, as transformadas Wavelets, dentre outras.

# 1- Codificação por transformada em blocos ...cont

Sistema típico de codificação por transformada em blocos:



- O decodificador implementa a sequência inversa de passos do codificador, com exceção da função de quantização.
- A compressão é alcançada durante a quantização dos coeficientes transformados, não durante o passo da transformação.
- A escolha de determinada transformada para uma dada aplicação depende da quantidade de erro de reconstrução que pode ser tolerada e dos recursos computacionais disponíveis.

# Compressão JPEG

- ❖ *Joint Photographic Experts Group* - nome do comitê que criou o padrão.
- ❖ Desenvolvido para possibilitar a compressão de imagens de sujeitos reais, tais como fotografias, tanto coloridas como em escala de cinza.
- ❖ Tem como característica intrínseca a perda de qualidade na compressão da imagem.
- ❖ Força a ocorrência de redundância para então eliminá-la.
- ❖ Uma imagem que sofre descompressão não é exatamente igual à original.
- ❖ Explora a redundância interpixel.
- ❖ Usa codificação por meio de transformada em blocos.
- ❖ Etapas para codificação de cada bloco da imagem:
  - Cálculo da Transformada Discreta do Cosseno (DCT)
  - Quantização
  - Atribuição do código de tamanho variável.


## Transformada Discreta do Cosseno (DCT)

- ✓ A DCT é aplicada a cada bloco da imagem.
- ✓ Apresenta a propriedade de compactação da energia em regiões próximas da origem.
- ✓ Muitos coeficientes apresentarão valores próximos a zero, podendo ser eliminados sem que haja perda significativa de informações contidas na imagem
- ✓ Antes da aplicação da DCT, a sub-imagem é transformada de uma escala positiva para uma centrada ao redor de zero.
- ✓ Para uma imagem de 8 bits cada *pixel* tem  $2^8 = 256$  valores de intensidade possíveis:  $[0, 255]$

- ✓ Para centrar ao redor de zero é necessário subtrair pela metade do número de valores possíveis, ou  $128 = (27)$ .
- ✓ Assim os valores dos *pixels* estarão na faixa de  $[-128, 127]$ .


## Transformada Discreta do Cosseno (DCT)

Suponha, como exemplo, que a matriz ao lado representa uma subimagem (bloco) 8x8 *pixels*, com *pixels* de 8 bits.



	0		...		7			
0	52	55	61	66	70	61	64	73
	63	59	55	90	109	85	69	72
	62	59	68	113	144	104	66	73
.	63	58	71	122	154	106	70	69
.	67	61	68	104	126	88	68	70
.	79	65	60	70	77	68	58	75
	85	71	64	59	55	61	65	83
7	87	79	69	68	65	76	78	94

Subtraímos então o valor 128 (deslocamento de nível) de cada pixel da subimagem, obtendo o arranjo deslocado ao lado.



-76	-73	-67	-62	-58	-67	-64	-55
-65	-69	-73	-38	-19	-43	-59	-56
-66	-69	-60	-15	16	-24	-62	-55
-65	-70	-57	-6	26	-22	-58	-59
-61	-67	-60	-24	-2	-40	-60	-58
-49	-63	-68	-58	-51	-60	-70	-53
-43	-57	-64	-69	-73	-67	-63	-45
-41	-49	-59	-60	-63	-52	-50	-34

## Transformada Discreta do Cosseno (DCT)

A DCT direta é dada pelas equações:

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

$$\alpha(k) = \begin{cases} \sqrt{1/N} & \text{para } k = 0 \\ \sqrt{2/N} & \text{para } k = 1, 2, \dots, N-1 \end{cases}$$

Aplicando estas equações da DCT na subimagem deslocada, com N=8, temos:

✓ Todos os coeficientes foram arredondados para o inteiro mais próximo.

✓ O valor mais alto da matriz, na parte superior esquerda é chamado de **coeficiente DC** → Representa a cor fundamental dos 64 pixels.

✓ Os 63 coeficientes restantes são chamados **coeficientes AC** → Representam os valores das pequenas variações de tonalidade ou cor.

-415	-29	-62	25	55	-20	-1	3
7	-21	-62	9	11	-7	-6	6
-46	8	77	-25	-30	10	7	-5
-50	13	35	-15	-9	6	0	3
11	-8	-13	-2	-1	1	-4	1
-10	1	3	-3	-1	0	2	-1
-4	-1	2	-1	2	-3	1	-2
-1	-1	-1	-2	-1	-1	0	-1



## Quantização

✓ Consiste na redução da informação dos componentes.

✓ Cada componente é dividido por uma constante para aquele componente, e então, são realizadas aproximações para o inteiro mais próximo.

✓ São utilizadas tabelas de quantização.

✓ Coeficientes menos importantes são descartados.

✓ Matriz de quantização típica:

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

## Quantização

- ✓ Cada coeficiente da matriz transformada deve ser ajustado e truncado de acordo com a equação:

$$T'(u,v) = \text{round}\left(\frac{T(u,v)}{Q(u,v)}\right)$$

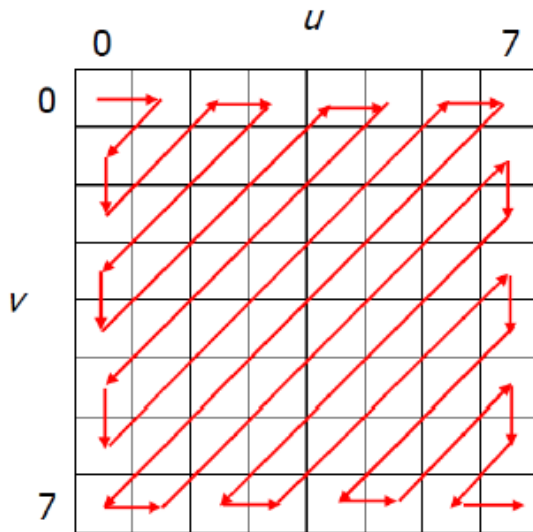
Se a matriz de quantização típica for utilizada para quantizar a matriz transformada do exemplo dado anteriormente, teremos:

Matriz transformada	Matriz de quantização típica	Resultado da quantização																																																																																																																																																																																																
<table><tr><td>-415</td><td>-29</td><td>-62</td><td>25</td><td>55</td><td>-20</td><td>-1</td><td>3</td></tr><tr><td>7</td><td>-21</td><td>-62</td><td>9</td><td>11</td><td>-7</td><td>-6</td><td>6</td></tr><tr><td>-46</td><td>8</td><td>77</td><td>-25</td><td>-30</td><td>10</td><td>7</td><td>-5</td></tr><tr><td>-50</td><td>13</td><td>35</td><td>-15</td><td>-9</td><td>6</td><td>0</td><td>3</td></tr><tr><td>11</td><td>-8</td><td>-13</td><td>-2</td><td>-1</td><td>1</td><td>-4</td><td>1</td></tr><tr><td>-10</td><td>1</td><td>3</td><td>-3</td><td>-1</td><td>0</td><td>2</td><td>-1</td></tr><tr><td>-4</td><td>-1</td><td>2</td><td>-1</td><td>2</td><td>-3</td><td>1</td><td>-2</td></tr><tr><td>-1</td><td>-1</td><td>-1</td><td>-2</td><td>-1</td><td>-1</td><td>0</td><td>-1</td></tr></table>	-415	-29	-62	25	55	-20	-1	3	7	-21	-62	9	11	-7	-6	6	-46	8	77	-25	-30	10	7	-5	-50	13	35	-15	-9	6	0	3	11	-8	-13	-2	-1	1	-4	1	-10	1	3	-3	-1	0	2	-1	-4	-1	2	-1	2	-3	1	-2	-1	-1	-1	-2	-1	-1	0	-1	<table><tr><td>16</td><td>11</td><td>10</td><td>16</td><td>24</td><td>40</td><td>51</td><td>61</td></tr><tr><td>12</td><td>12</td><td>14</td><td>19</td><td>26</td><td>58</td><td>60</td><td>55</td></tr><tr><td>14</td><td>13</td><td>16</td><td>24</td><td>40</td><td>57</td><td>69</td><td>56</td></tr><tr><td>14</td><td>17</td><td>22</td><td>29</td><td>51</td><td>87</td><td>80</td><td>62</td></tr><tr><td>18</td><td>22</td><td>37</td><td>56</td><td>68</td><td>109</td><td>103</td><td>77</td></tr><tr><td>24</td><td>35</td><td>55</td><td>64</td><td>81</td><td>104</td><td>113</td><td>92</td></tr><tr><td>49</td><td>64</td><td>78</td><td>87</td><td>103</td><td>121</td><td>120</td><td>101</td></tr><tr><td>72</td><td>92</td><td>95</td><td>98</td><td>112</td><td>100</td><td>103</td><td>99</td></tr></table>	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	<table><tr><td>-26</td><td>-3</td><td>-6</td><td>2</td><td>2</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>-2</td><td>-4</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>-3</td><td>1</td><td>5</td><td>-1</td><td>-1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>-4</td><td>1</td><td>2</td><td>-1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	-26	-3	-6	2	2	0	0	0	1	-2	-4	0	0	0	0	0	-3	1	5	-1	-1	0	0	0	-4	1	2	-1	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
-415	-29	-62	25	55	-20	-1	3																																																																																																																																																																																											
7	-21	-62	9	11	-7	-6	6																																																																																																																																																																																											
-46	8	77	-25	-30	10	7	-5																																																																																																																																																																																											
-50	13	35	-15	-9	6	0	3																																																																																																																																																																																											
11	-8	-13	-2	-1	1	-4	1																																																																																																																																																																																											
-10	1	3	-3	-1	0	2	-1																																																																																																																																																																																											
-4	-1	2	-1	2	-3	1	-2																																																																																																																																																																																											
-1	-1	-1	-2	-1	-1	0	-1																																																																																																																																																																																											
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																																																																																																																																																																																											
-26	-3	-6	2	2	0	0	0																																																																																																																																																																																											
1	-2	-4	0	0	0	0	0																																																																																																																																																																																											
-3	1	5	-1	-1	0	0	0																																																																																																																																																																																											
-4	1	2	-1	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																																																																																																																																																																																											

Ex: Coeficiente DC  $\rightarrow T'(0,0) = \text{round}\left[\frac{T(0,0)}{Q(0,0)}\right] = \text{round}\left[\frac{-415}{16}\right] = -26$

- ✓ Para realizar a codificação são executados 3 passos principais:
- ✓ Ordenação em zigue-zague.
- ✓ Codificação *run-length* (RLE).
- ✓ Codificação entrópica (Huffman).

Após a quantização, os coeficientes são colocados em um array 1D por ordem de frequência, seguindo a sequência em zigue-zague ilustrada abaixo:



Fazendo a ordenação da matriz quantizada do exemplo anterior, temos o vetor:

- 26, -3, 1, -3, -2, -6,  
2, -4, 1, -4, 1, 1,  
5, 0, 2, 0, 0, -1, 2, 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, 0, 0, 0,

## Matriz quantizada

-26	-3	-6	2	2	0	0	0
1	-2	-4	0	0	0	0	0
-3	1	5	-1	-1	0	0	0
-4	1	2	-1	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

## Codificação

## Codificação *run-length* (RLE):

Abaixo, é ilustrado como se dá a codificação run-length.

O objetivo desta codificação é comprimir sequências de zero.

## Vetor 1D ordenado:

[illegible]

### Resultado da codificação RLE:

Obs: O coeficiente DC não é codificado aqui.

Número de zeros anteriores

1º. coeficiente

Valor do coeficiente

[	0/-3	0/1	0/-3	0/-2	0/-6	0/2	0/-4
0/1	0/-4	0/1	0/1	0/5	1/2	2/-1	0/2
5/-1	0/-1	EOB	]				

Símbolo

Condição de fim de bloco (*end of block*)

## Codificação

### Codificação entrópica (Huffman):

- Gera códigos de comprimento variável.
- Útil para aumentar a compressão.
- Utiliza tabelas de codificação de Huffman predefinidas.
- O código binário tem a forma  $\rightarrow < \text{PREFIXO} : \text{MANTISSA} >$
- O PREFIXO e a MANTISSA têm comprimento variável.

### Codificação do coeficiente DC:

Calcula-se e codifica-se a diferença entre o coeficiente DC atual e aquele da subimagem previamente codificada.

No nosso exemplo suponha que o coeficiente DC da subimagem imediatamente à esquerda seja -17. A diferença será:  $[-26 - (-17)] = -9$

**PREFIXO:** Definido pela categoria (ordem de grandeza) da diferença.  
Obtém-se a categoria pela Tabela 1 e, a partir da categoria, o código é obtido diretamente pela Tabela 2.

**MANTISSA:** Define o valor do coeficiente e o seu sinal.

Ver exemplo no slide 56.

# Compressão JPEG

...cont

## Codificação

### Codificação entrópica (Huffman):

**TABELA 1**

Faixa de Valores	Categoria
0	0
-1; +1	1
-3, -2; +2, +3	2
-7 a -4; +4 a +7	3
-15 a -8; +8 a +15	4
-31 a -16; +16 a +31	5
-63 a -32; +32 a +63	6
-127 a -64; +64 a +127	7
-255 a -128; +128 a +255	8
-511 a -256; +256 a +511	9
-1023 a -512; +512 a +1023	A
-2047 a -1024; +1024 a +2047	B

# Compressão JPEG

...cont

## Codificação

Codificação entrópica (Huffman):

**TABELA 2 – Codificação DC**

<b>Categoria</b>	<b>Prefixo</b>	<b>Comprimento total (bits)</b>	<b>Comprimento Mantissa</b>
<b>0</b>	010	<b>3</b>	<b>0</b>
<b>1</b>	011	<b>4</b>	<b>1</b>
<b>2</b>	100	<b>5</b>	<b>2</b>
<b>3</b>	00	<b>5</b>	<b>3</b>
<b>4</b>	101	<b>7</b>	<b>4</b>
<b>5</b>	110	<b>8</b>	<b>5</b>
<b>6</b>	1110	<b>10</b>	<b>6</b>
<b>7</b>	11110	<b>12</b>	<b>7</b>
<b>8</b>	111110	<b>14</b>	<b>8</b>
<b>9</b>	1111110	<b>16</b>	<b>9</b>
<b>A</b>	11111110	<b>18</b>	<b>10</b>
<b>B</b>	111111110	<b>20</b>	<b>11</b>

## Codificação

## Exemplo da codificação do coeficiente DC:

- Definição do **prefixo**:
  - **Diferença** do coeficiente DC = -9
  - Pela Tabela 1, este valor é representado pela categoria 4 (faixa de valores: -15 a -8 e 8 a 15)
  - Pela Tabela 2, o **prefixo** correspondente é **101**
  - Como o comprimento total do código deve ser 7 bits, a mantissa terá 4 bits.
- Definição da **mantissa**: Diferencia os coeficientes representados pela mesma categoria.
  - Existem 16 coeficientes representados pela categoria 4. Esses coeficientes podem ser perfeitamente diferenciados através de códigos de 4 bits, conforme a tabela abaixo:

Coef.	Código	Coef.	Código	Coef.	Código	Coef.	Código
-15	0000	-11	0100	8	1000	12	1100
-14	0001	-10	0101	9	1001	13	1101
-13	0010	-9	0110	10	1010	14	1110
-12	0011	-8	0111	11	1011	15	1111

Coeficiente DC completamente codificado: **1 0 1 0 1 1 0**



## Codificação

## Codificação dos coeficiente AC:

Os coeficientes AC não nulos do vetor reordenado devem ser codificados na forma <PREFIXO:MANTISSA>.

**PREFIXO:** Com a informação do número de zeros antes do coeficiente e da categoria do coeficiente, busca-se na Tabela 3 o código do prefixo.

**MANTISSA:** Define o valor do coeficiente e o seu sinal.  
É obtida da mesma forma que para o coeficiente DC.

**Exemplo:** O primeiro coeficiente não nulo do vetor reordenado é (-3)

- Definição do **prefixo**:
  - Este coeficiente não foi precedido por zeros.
  - Pela Tabela 1, este valor é representado pela categoria 2 (faixa: -3 a -2 e 2 a 3)
  - Pela Tabela 3, o **prefixo** correspondente é **01**
  - Como o comprimento total do código deve ser 4 bits, a mantissa terá 2 bits.
- Definição da **mantissa**: Pela tabela abaixo a mantissa será **00**

Coef	Código	Coef.	Código	Codificação completa do coeficiente (-3):
-3	00	2	10	
-2	01	3	11	<b>0 1 0 0</b>

## Codificação entrópica (Huffman):

**TABELA 3 – Codificação AC**

Zeros/ Categoria	Prefixo	Comprim. total
0/1	00	3
0/2	01	4
0/3	100	6
0/4	1011	8
0/5	11010	10
0/6	111000	12
0/7	1111000	14
0/8	111110110	18
0/9	111111110000010	25
0/A	1111111110000011	26
1/1	1100	5
1/2	111001	8
1/3	1111001	10

Zeros/ Categoria	Prefixo	Comprim. total
3/1	111010	7
3/2	111110111	11
3/3	11111110111	14
3/4	1111111110010000	20
...		
13/1	11111111010	12
13/2	1111111111100011	18
13/3	1111111111100100	19
...		
15/7	1111111111111011	23
15/8	1111111111111100	24
15/9	1111111111111101	25
15/A	1111111111111110	26

## Codificação

- Como visto, o símbolo EOB representa a condição de fim de bloco.

Uma palavra-código de Huffman especial para o EOB é atribuída para indicar que o restante dos coeficientes em uma sequência reordenada é nulo.

EOB → 1 0 1 0

Vetor reordenado completamente codificado:

-26 →      0/2/-3      0/1/1  
 1010110 0100 001 0100 0101 100001 0110  
 100011 001 100011 001 001 100101 11100110  
 110110 0110 11110100 000 1010  
 EOB

Bloco original:  $8 \times 8 \times 8 = 512$  bits; Bloco comprimido: 92 bits  
 (Compressão: 5,6 : 1)

# Descompressão JPEG

- Para descomprimir uma subimagem comprimida por JPEG, o decodificador deve primeiro recriar os coeficientes da transformada quantizada que levaram à cadeia de bits comprimida.
- Como uma sequência binária codificada por Huffman é decodificável instantaneamente de forma única, esse passo é facilmente realizado por meio de uma simples tabela indexada (*lookup-table*).

1) A matriz novamente gerada dos coeficientes quantizados é:

-26	-3	-6	2	2	0	0	0
1	-2	-4	0	0	0	0	0
-3	1	5	-1	-1	0	0	0
-4	1	2	-1	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

2) A desnormalização através da equação:

$$\hat{T}(u, v) = T'(u, v) Q(u, v)$$

Resulta em:

-416	-33	-60	32	48	0	0	0
12	-24	-56	0	0	0	0	0
-42	13	80	-24	-40	0	0	0
-56	17	44	-29	0	0	0	0
18	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

# Descompressão JPEG

3) Tomando-se a DCT inversa da matriz desnormalizada, obtem-se:

-70	-64	-61	-64	-69	-66	-58	-50
-72	-73	-61	-39	-30	-40	-54	-59
-68	-78	-58	-9	13	-12	-48	-64
-59	-77	-57	0	22	-13	-51	-60
-54	-75	-64	-23	-13	-44	-63	-56
-52	-71	-72	-54	-54	-71	-71	-54
-45	-59	-70	-68	-67	-67	-61	-50
-35	-47	-61	-66	-60	-48	-44	-44

4) Deslocando o nível de cada pixel inversamente transformado por  $+128 = (+27)$ , obtem-se:

58	64	67	64	59	62	70	78
56	55	67	89	98	88	74	69
60	50	70	119	141	116	80	64
69	51	71	128	149	115	77	68
74	53	64	105	115	84	65	72
76	57	56	74	75	57	57	74
83	69	59	60	61	61	67	78
93	81	67	62	69	80	84	84

## Erro de reconstrução:

-6	-9	-6	2	11	-1	-6	-5
7	4	-1	1	11	-3	-5	3
2	9	-2	-6	-3	-12	-14	9
-6	7	0	-4	-5	-9	-7	1
-7	8	4	-1	6	4	3	-2
3	8	4	-4	2	6	1	1
2	2	5	-1	-6	0	-2	5
-6	-2	2	6	-4	-4	-6	10

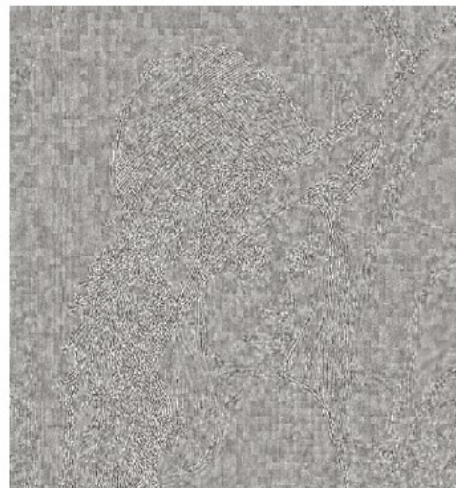
O erro de raiz média quadrática do processo completo de compressão e reconstrução é 5,8 níveis de cinza.

# Exemplo: Ilustração da codificação JPEG

Imagens com  
codificação JPEG



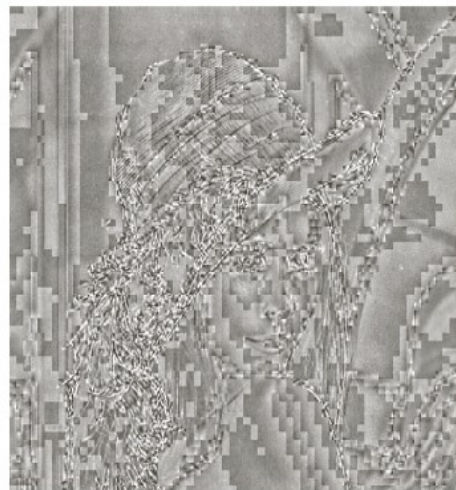
Diferença entre a  
imagem original e as  
imagens reconstruídas



Área ampliada da  
imagem reconstruída



Compressão = 25:1  
Erro rms = 5,4



Compressão = 52:1  
Erro rms = 10,7

## Compressão JPEG - 2000

- ❖ Estende o popular JPEG para proporcionar maior flexibilidade tanto à compressão de imagens estáticas de um tom contínuo quanto ao acesso aos dados comprimidos.
- ❖ Por exemplo, partes de uma imagem comprimida pelo padrão JPEG-2000 podem ser extraídas para a retransmissão, armazenamento, exibição e/ou edição.
- ❖ O padrão baseia-se nas técnicas de compressão wavelet, utilizando a **transformada wavelet discreta**.
- ❖ A quantização de coeficientes é adaptada a escalas e sub-bandas individuais e os coeficientes quantizados são codificados utilizando a **codificação aritmética**.



# Exemplo: Comparação entre a codificação JPEG-2000 e a codificação JPEG

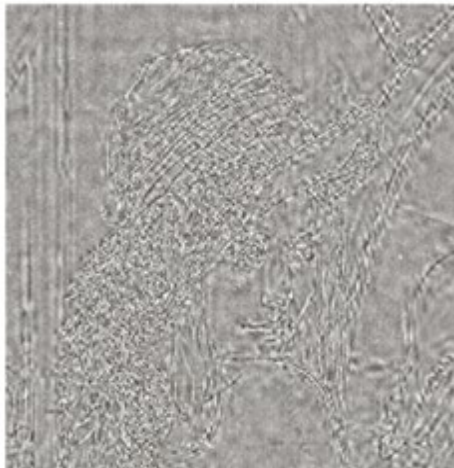
Imagens com  
codificação  
JPEG-2000

Diferença entre a  
imagem original e as  
imagens reconstruídas

Área ampliada da  
imagem reconstruída



Compressão = 25:1  
Erro rms = 3,86



Compressão = 52:1  
Erro rms = 5,77



**Exemplo:** Comparação entre a codificação JPEG-2000 e a codificação JPEG



JPEG



JPEG2000

## Codificação de vídeo M-JPEG e MPEG

- Apesar de JPEG ter sido concebido para compressão de imagens estáticas, muitos fabricantes aplicaram o JPEG para sequências de imagens de vídeo ( M-JPEG )
- Falta de padrão ocasiona diferentes implementações
- O Padrão reconhecido é o MPEG ( Moving pictures Experts Group )  
comprime quadros isolados em JPEG e utiliza redundância inter frame
- MPEG tem taxa de compressão cerca de 3 vezes superior e M-JPEG
- A compressão inter-frame dificulta a edição de vídeo o que favorece a popularidade de M-JPEG nos equipamentos de edição de vídeo

**FIM**