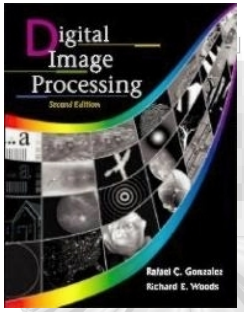
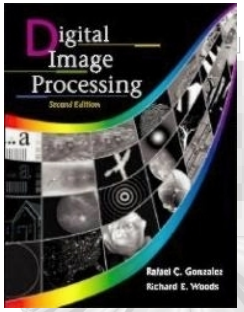



# Aula 9.3

## Compressão de Imagens

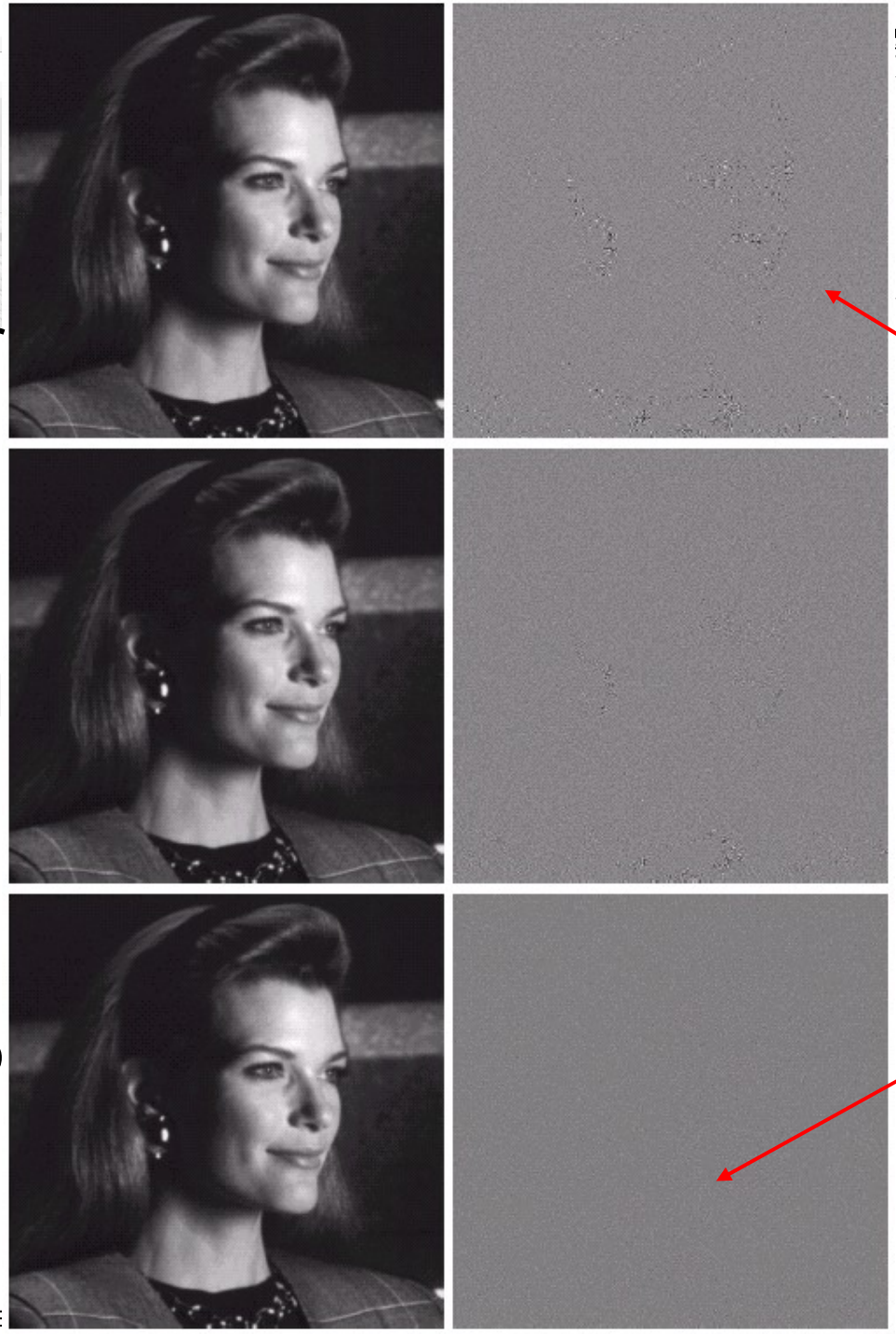



# Image Compression

Fourier

Hadamard

Cosseno

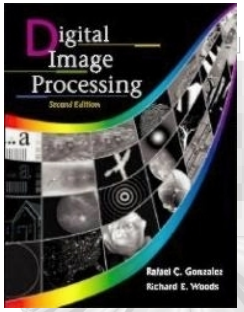


Erros obtidos

**Erros Maiores**

**Erros Menores**

A maioria dos sistemas práticos usa a DCT

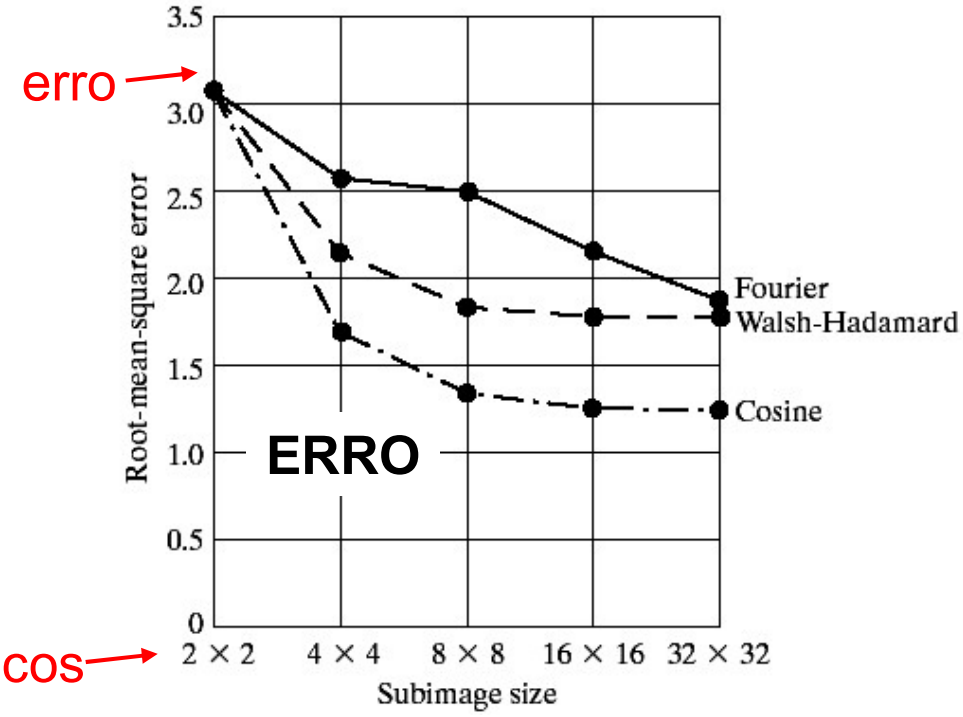


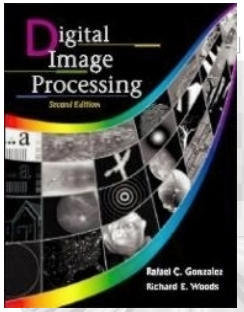

# Image Compression

## Seleção do tamanho da subimagem

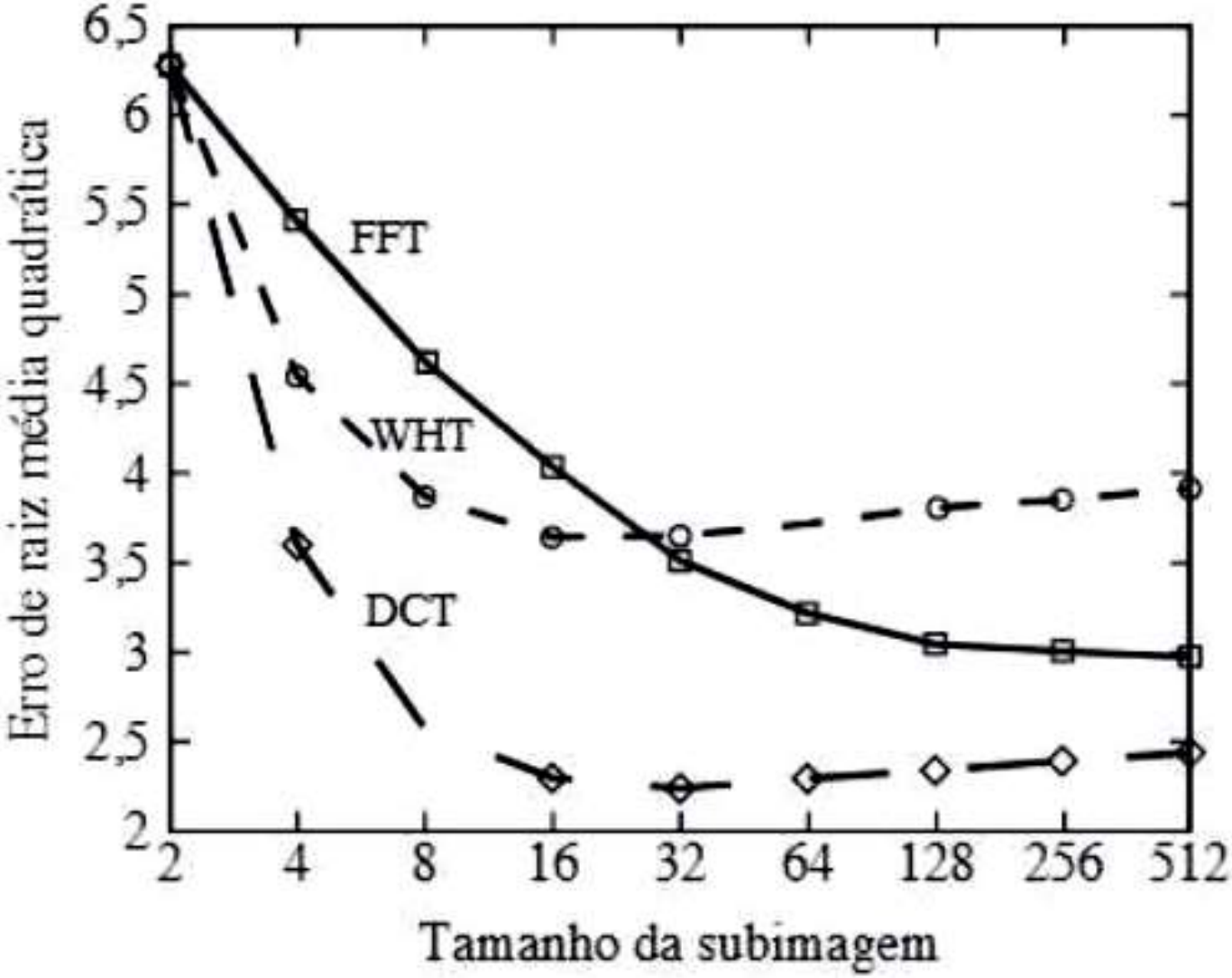
O processamento de imagens muito grandes se torna muito complexo computacionalmente, por isto, geralmente, as imagens são divididas em **blocos** (subimagens) e processadas individualmente

Além disso, o tamanho das subimagens é um fator significante, que afeta o erro

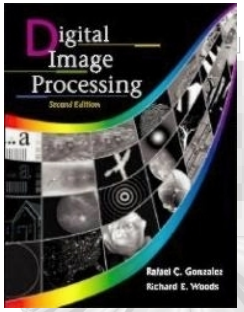





# Image Compression




Erro de reconstrução *versus* tamanho da subimagem.

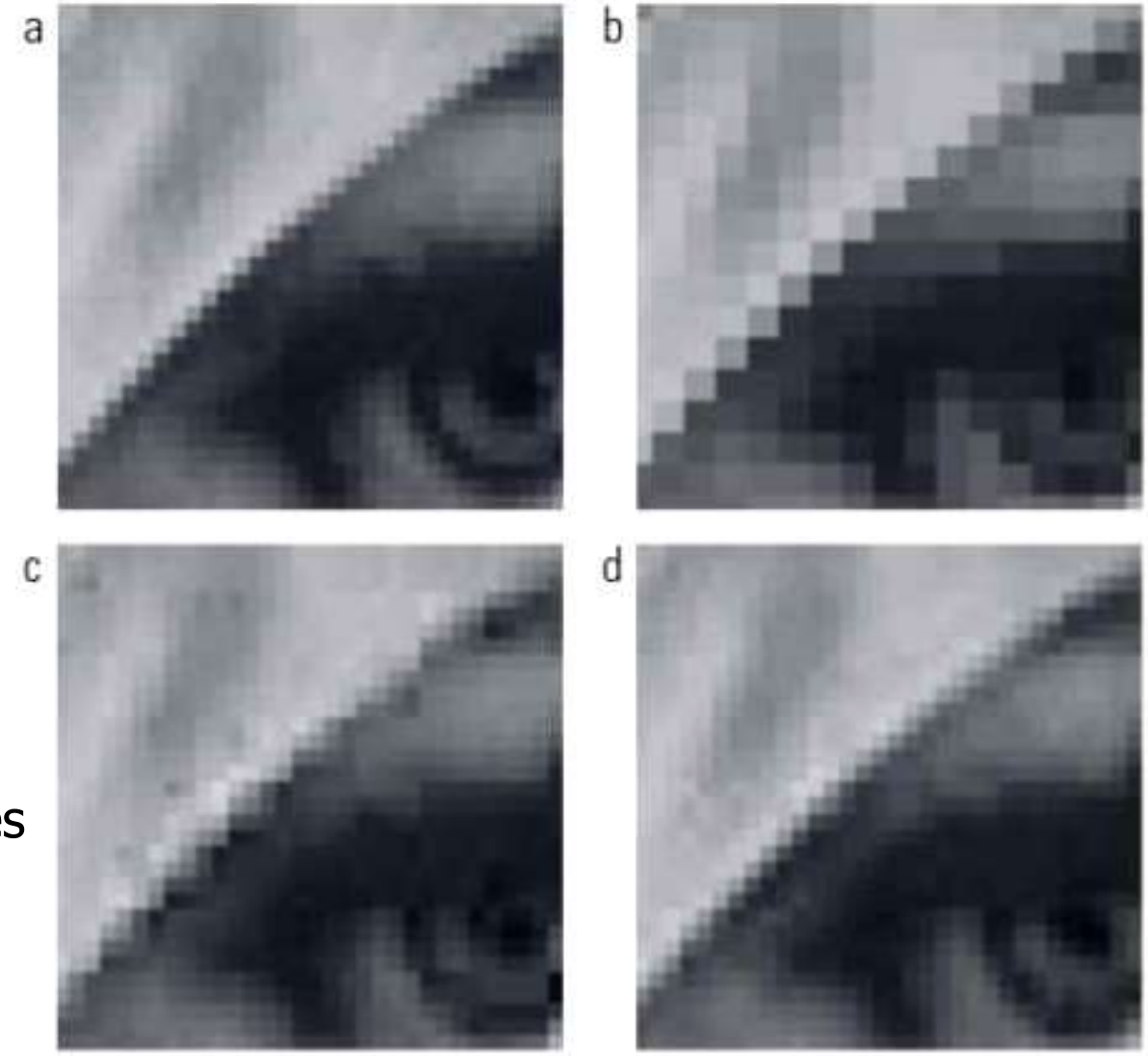



# Image Compression

## Seleção do tamanho da subimagem

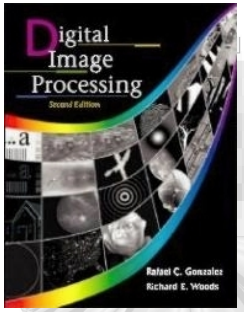
Original

a) Zoom da original
b) Aproximações de (a) usando 25% dos coeficientes da DCT e blocos 2x2
a) blocos 4x4
b) blocos 8x8



# Image Compression

A transformada do Cosseno é a mais usada na prática, sendo empregada no formato JPEG, MPEG, etc.




# Image Compression

**TDC**

$$C(i,j) = \alpha(i)\alpha(j)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left(\frac{(2x+1)i\pi}{2N}\right)\cos\left(\frac{(2y+1)j\pi}{2N}\right)$$

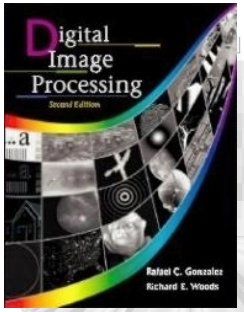
$$\alpha(x) = \begin{cases} \dfrac{1}{\sqrt{N}} & \text{se} \quad x = 0 \\ \sqrt{\dfrac{2}{\mathrm{N}}} & \text{para} \quad x = 1, 2, ..., \mathrm{N}-1 \end{cases}$$

**ITDC**

$$f(x,y) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} \alpha(i)\alpha(j)C(i,j)\cos\left(\frac{(2x+1)i\pi}{2N}\right)\cos\left(\frac{(2y+1)j\pi}{2N}\right)$$

**Codificação por Zona** - Como as transformadas tendem a manter a maior parte da informação nos coeficientes próximos de 0,0, são usadas funções de mascaramento, como:

$$\chi(u,v) = \begin{cases} 0 & \text{se } T(u,v) \text{ satisfaz um critério de} \\ & \text{truncamento específico} \\ 1 & \text{caso contrário} \end{cases}$$




$$\chi(u,v) = \begin{cases} 0 & \text{se } T(u,v) \text{ satisfaz um critério de} \\ & \text{truncamento específico} \\ 1 & \text{caso contrário} \end{cases}$$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 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 |

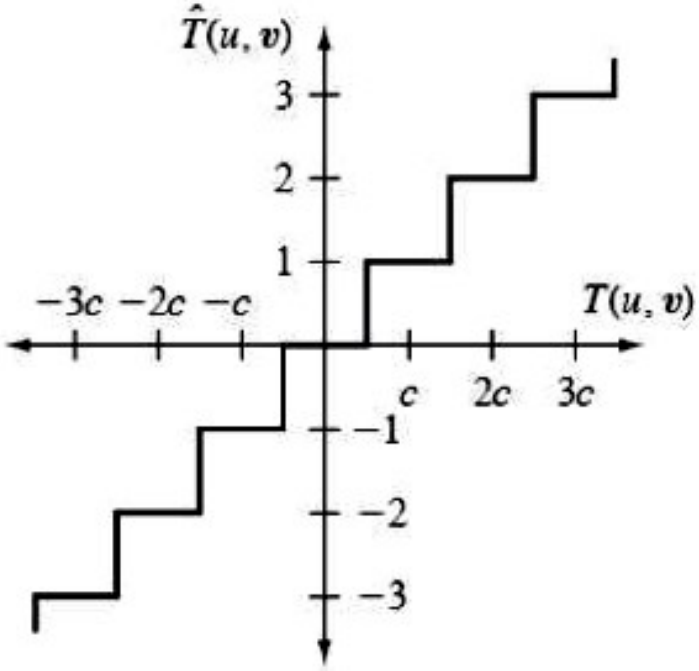
zig-zag

**Codificação por Limiarização** – Enquanto a codificação por zonas costuma ser implementada por meio de uma única máscara fixa para todas as subimagens, a codificação por limiarização é adaptativa, pois a posição dos coeficientes da transformada a serem preservados varia de uma subimagem para outra

Há três formas básicas de limiarizar uma subimagem:

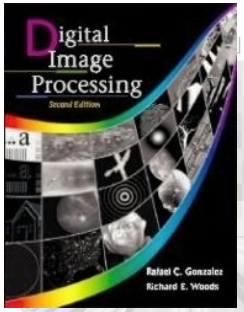
1) Um único limiar global, a ser aplicado em todas as subimagens
2) Um limiar diferente para cada subimagem
3) O limiar pode variar em função da posição de cada coeficiente dentro da subimagem

# Exemplo em que o limiar pode varia em função da posição de cada coeficiente dentro da subimagem




| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 |

Matriz de Limiarização usada no JPEG - pondera cada coeficiente de uma subimagem transformada, de acordo com a sua importância perceptual e psicovisual



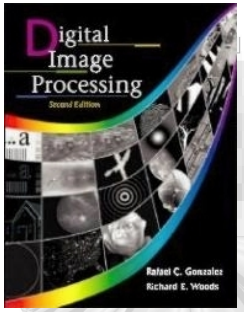

# Codificação e decodificação Baseline JPEG

Seja a subimagem 8x8

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 52 | 55 | 61 | 66 | 70 | 61 | 64 | 73 |
| 63 | 59 | 66 | 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 | 63 | 58 | 75 |
| 85 | 71 | 64 | 59 | 55 | 61 | 65 | 83 |
| 87 | 79 | 69 | 68 | 65 | 76 | 78 | 94 |

-128

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −76 | −73 | −67 | −62 | −58 | −67 | −64 | −55 |
| −65 | −69 | −62 | −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 | −65 | −70 | −53 |
| −43 | −57 | −64 | −69 | −73 | −67 | −63 | −45 |
| −41 | −49 | −59 | −60 | −63 | −52 | −50 | −34 |




# Codificação e decodificação Baseline JPEG

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −76 | −73 | −67 | −62 | −58 | −67 | −64 | −55 |
| −65 | −69 | −62 | −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 | −65 | −70 | −53 |
| −43 | −57 | −64 | −69 | −73 | −67 | −63 | −45 |
| −41 | −49 | −59 | −60 | −63 | −52 | −50 | −34 |

DCT

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −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 |

# Codificação e decodificação Baseline JPEG

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −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 |

## exemplo

$$\hat{T}(0,0) = \text{arred}\left[\frac{T(0,0)}{Z(0,0)}\right]$$

$$= \text{arred}\left[\frac{-415}{16}\right] = -26$$

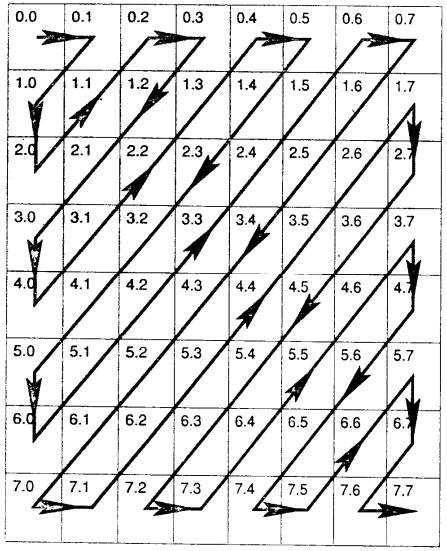
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 16 | 11 | 10 | 16 | 24 | 40 | 51 | 61 |
| 12 | 12 | 14 | 19 | 26 | 58 | 60 | 55 |
| 14 | 13 | 16 | 24 | 40 | 57 | 69 | 56 |
| 14 | 17 | 22 | 29 | 51 | 87 | 80 | 62 |
| 18 | 22 | 37 | 56 | 68 | 109 | 103 | 77 |
| 24 | 35 | 55 | 64 | 81 | 104 | 113 | 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 |

# Codificação e decodificação Baseline JPEG

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −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 |

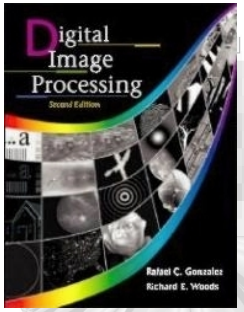
Zig-Zag




Zig-Zag

[−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 EOB]

Em seguida, codifica, usando Huffman com uma tabela de códigos fixa




# Codificação e decodificação Baseline JPEG

## Decodificação

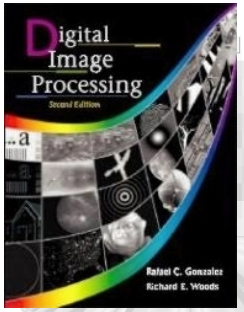
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −26 | −3 | −6 | 2 | 2 | 0 | 0 | 0 |
| 1 | −2 | −4 | 0 | 0 | 0 | 0 | 0 |
| −3 | 1 | 5 | −1 | −1 | 0 | 0 | 0 |
| −4 | 1 | 2 | −1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |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

Monta a matriz a partir dos valores no arquivo

$$\dot{T}(0,0) = \hat{T}(0,0)Z(0,0) = (-26)(16) = -416$$

Desquantiza

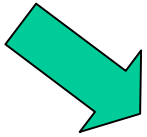
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −416 | −33 | −60 | 32 | 48 | 0 | 0 | 0 |
| 12 | −24 | −56 | 0 | 0 | 0 | 0 | 0 |
| −42 | 13 | 80 | −24 | −40 | 0 | 0 | 0 |
| −56 | 17 | 44 | −29 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0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




# Codificação e decodificação Baseline JPEG

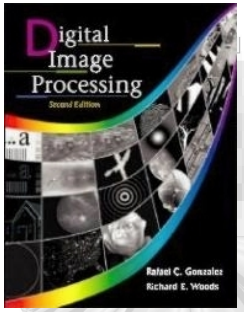
## Decodificação

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −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 |



IDCT

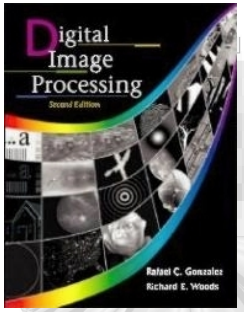
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −70 | −64 | −61 | −64 | −69 | −66 | −58 | −50 |
| −72 | −73 | −61 | −39 | −30 | −40 | −54 | −59 |
| −68 | −78 | −58 | −9 | 13 | −12 | −48 | −64 |
| −59 | −77 | −57 | 0 | 22 | −13 | −51 | −60 |
| −54 | −75 | −64 | −23 | −13 | −44 | −63 | −56 |
| −52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




| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −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 |

+128

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 |




## Original

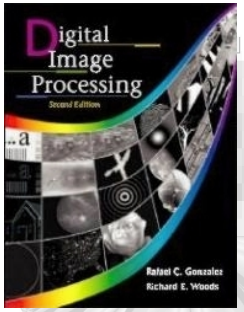
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 52 | 55 | 61 | 66 | 70 | 61 | 64 | 73 |
| 63 | 59 | 66 | 90 | 109 | 85 | 69 | 72 |
| 62 | 59 | 68 | 113 | 144 | 104 | 66 | 73 |
| 63 | 58 | 71 | 122 | 154 | 106 | 70 | 69 |
| 67 | 61 | 68 | 104 | 126 | 88 | 68 | 70 |
| 79 | 65 | 60 | 70 | 77 | 63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

## Decodificada

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 |

## Original – Decodificada =

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| −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 |




# Image Compression

## Exemplo: compressão usando a DCT

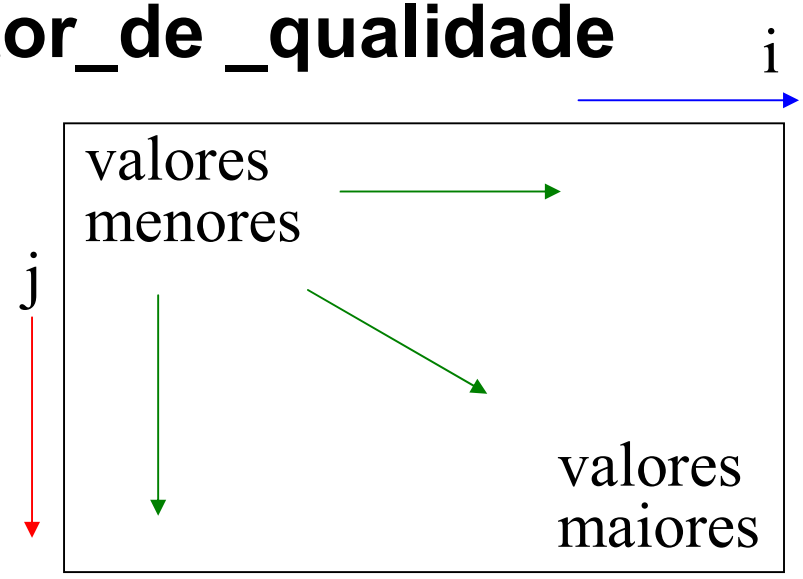
## QUANTIZAÇÃO

A TDC ocupa  mais espaço do que a matriz original, pois os coeficientes da TDC são reais;
Quantização: processo de reduzir o número de bits;
É neste processo que acontece a perda da informação;
No padrão JPEG:

$$Q[i][j] = 1 + (1 + i + j) * fator\_de\_qualidade$$

Fator_de_qualidade: 1 a 25

$$Valor\_quantizado\ [i][j] = \frac{DCT[i][j]}{Q[i][j]}$$

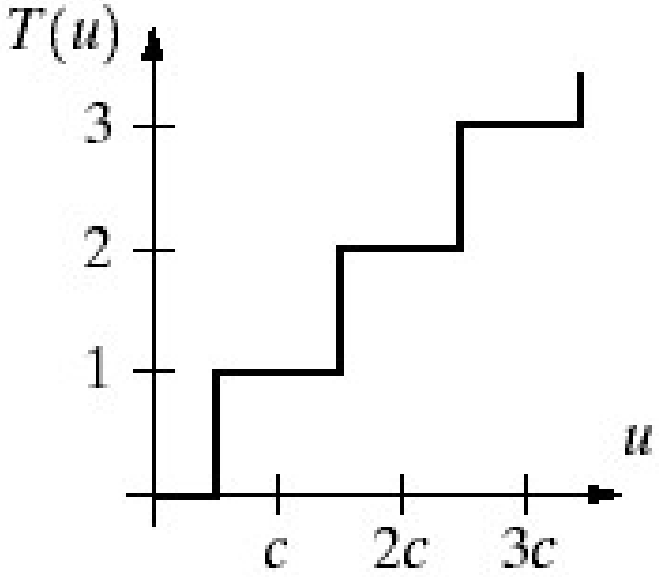


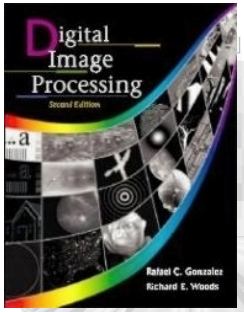
# Image Compression

## Curva de quantização

São usadas para obter os valores que dividirão os coeficientes, com o objetivo de reduzir e até eliminá-los

Em geral, os coeficientes menores são aqueles associados às altas-frequências da imagem, e que correspondem aos pequenos detalhes, que se omitidos, não causam muitas perdas.






# Matriz de quantização

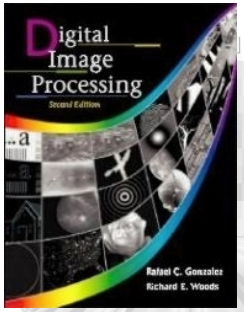
Image Compression

obtida com a função

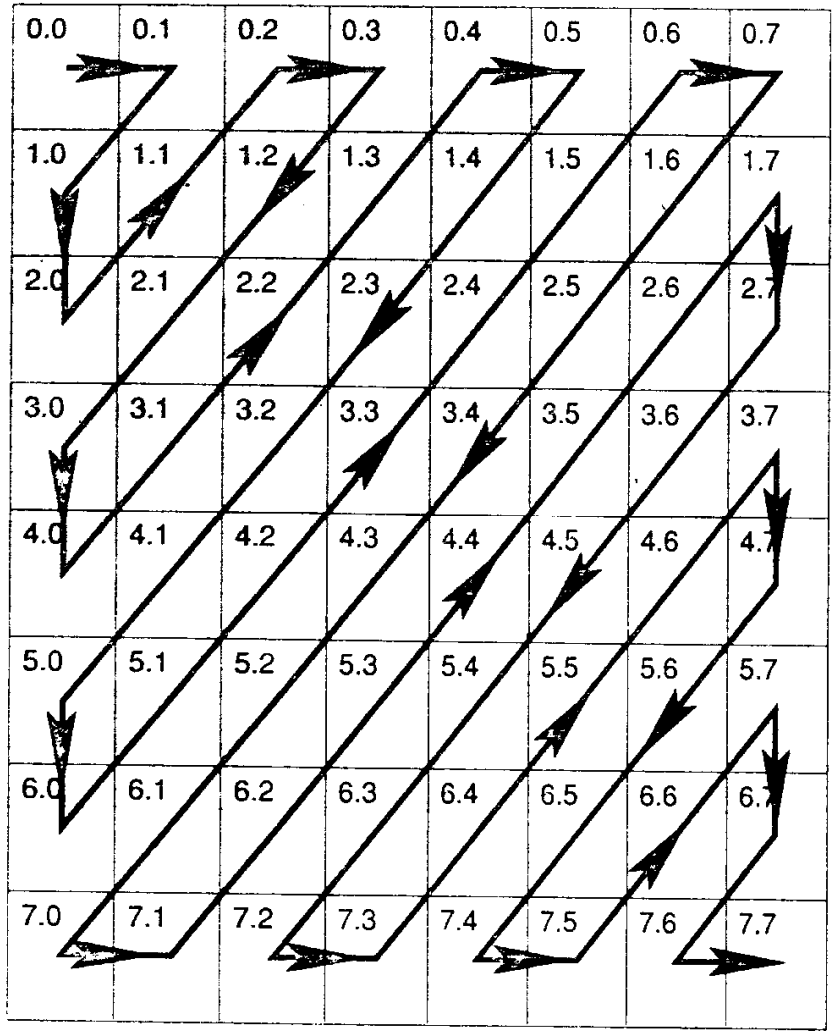
$$Q[i][j] = 1 + (1 + i + j) * fator\_de\_qualidade$$

**Ex.**
fator_de _qualidade = 2

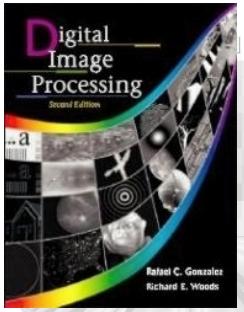
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 |
| 1 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 |
| 2 | 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 |
| 3 | 9 | 11 | 13 | 15 | 17 | 19 | 21 | 23 |
| 4 | 11 | 13 | 15 | 17 | 19 | 21 | 23 | 25 |
| 5 | 13 | 15 | 17 | 19 | 21 | 23 | 25 | 27 |
| 6 | 15 | 17 | 19 | 21 | 23 | 25 | 27 | 29 |
| 7 | 17 | 19 | 21 | 23 | 25 | 27 | 29 | 31 |
| | 19 | 21 | 23 | 25 | 27 | 29 | 31 | 33 |
| | 21 | 23 | 25 | 27 | 29 | 31 | 33 | 35 |




# Image Compression




Esquema zig-zag

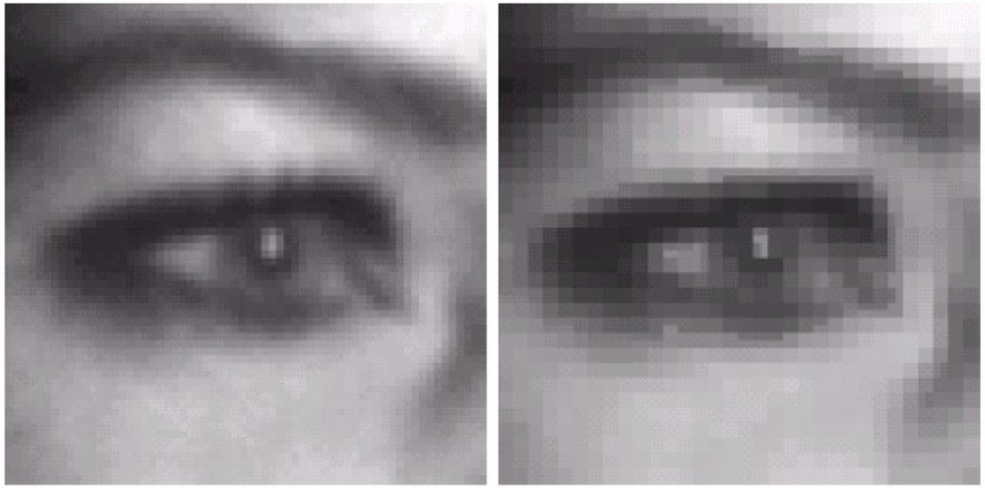





# Image Compression

## Usando apenas 25% dos coeficientes da DCT

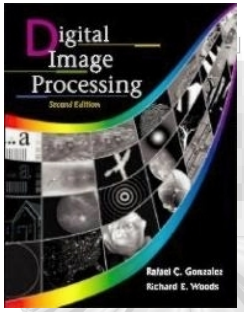
Original ampliada



Bloco 2x2

Bloco 4x4

Bloco 8x8






# Image Compression

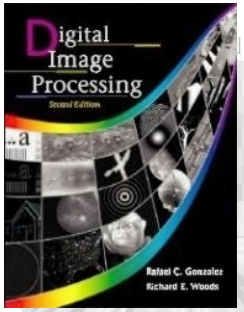
## Exemplo: compressão usando a DCT

## Armazenamento e transmissão

São armazenados ou transmitidos apenas os coeficientes que possuem outros coeficientes diferentes de zero adiante.

No processo inverso, estes coeficientes são recuperados e colocados em suas respectivas posições, usando o zig-zag.
Em seguida é aplicada a desquantização
E a transformada inversa
Com certeza ocorrem erros, mas, em geral, são aceitáveis

O padrão JPEG ainda inclui a codificação de Huffman no processo




# Image Compression

## Exemplo: compressão usando a DCT

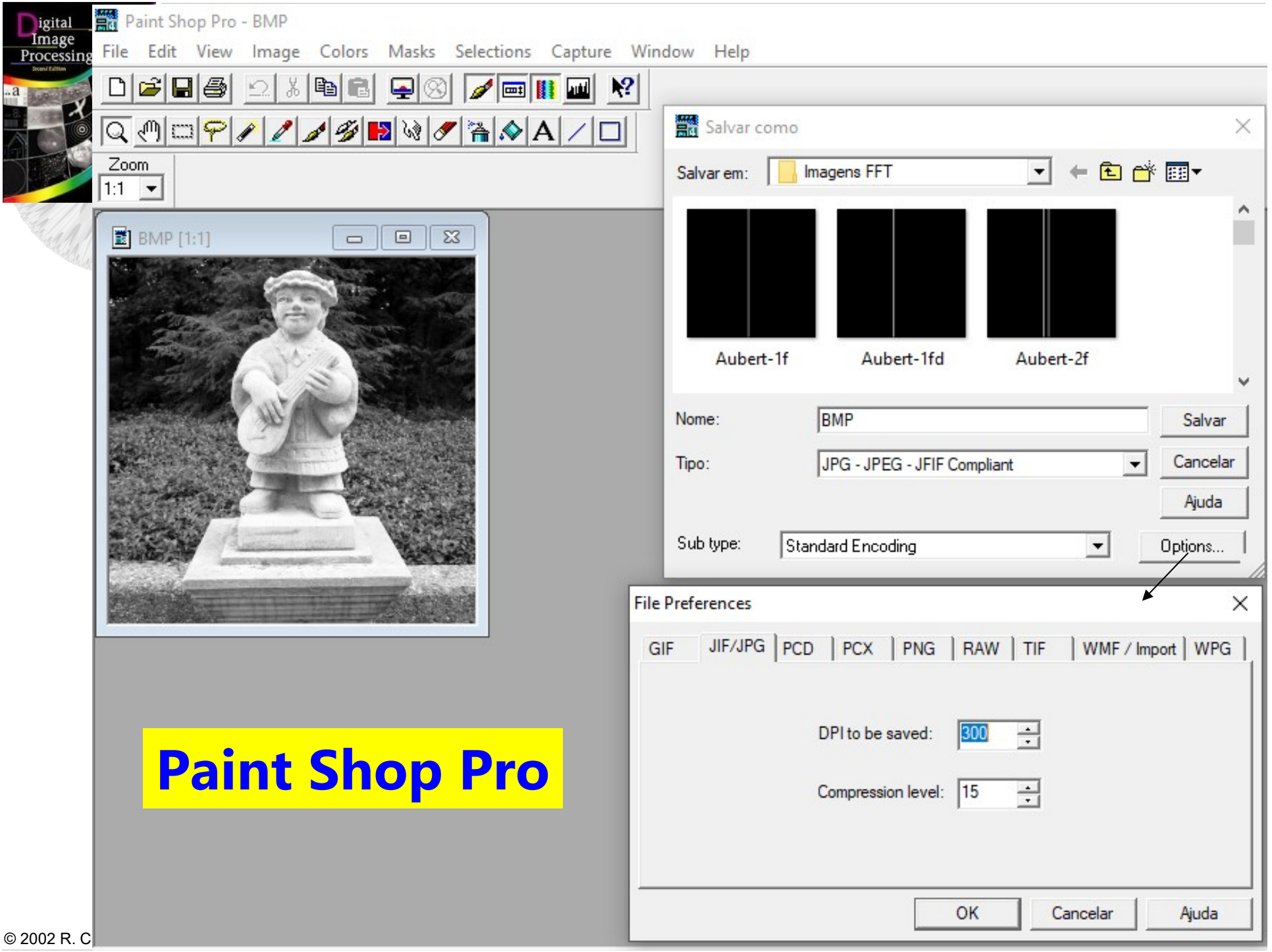
ruim

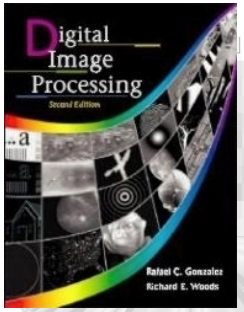
Imagem original

Fator de qualidade 10

Fator de qualidade 2

bom


Digital Image Processing
Paint Shop Pro - BMP
File Edit View Image Colors Masks Selections Capture Window Help
Zoom
1:1
BMP [1:1]
Salvar como
Salvar em:
Imagens FFT
Aubert-1f
Aubert-1fd
Aubert-2f
Nome:
BMP
Salvar
Tipo:
JPG - JPEG - JFIF Compliant
Cancelar
Ajuda
Sub type:
Standard Encoding
Options...
File Preferences
GIF
JIF/JPG
PCD
PCX
PNG
RAW
TIF
WMF / Import
WPG
DPI to be saved:
300
Compression level:
15
OK
Cancelar
Ajuda
Paint Shop Pro




(BMP)193kbytes

JPG(1)58kbytes

JPG(15)24kbytes

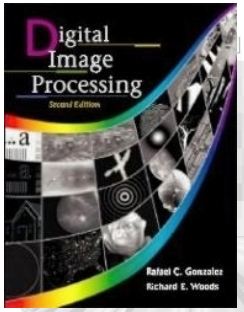
Compression Level

Tamanho do arquivo

JPG(50)12kbytes

JPG(50)12kbytes

JPG(99)3kbytes




# Compressão de vídeos

- As técnicas usadas com imagens, foram adaptadas para fazer a compressão de vídeos, que consideram as modificações de tons e cores nos frames (imagens), de um pixel para o outro e, também, de um frame para o outro

- **exemplo: JPEG para imagens**

  **MPEG para vídeos**

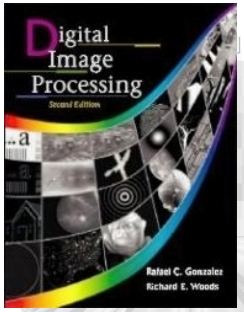


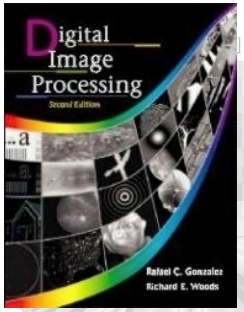

Image Compression

# Prática para entregar

1) Suponha a imagem 8x8 em que cada valor representa um tom de cinza (0→15), ou seja 4 bits por pixel

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 6 | 6 | 6 |
| 0 | 0 | 0 | 0 | 1 | 5 | 5 | 5 |
| 0 | 0 | 0 | 0 | 3 | 3 | 3 | 3 |
| 2 | 2 | 2 | 2 | 3 | 3 | 4 | 4 |
| 4 | 7 | 7 | 7 | 8 | 8 | 9 | 9 |
| 7 | 7 | 7 | 7 | 8 | 8 | 9 | 9 |
| 7 | 7 | 7 | 7 | 10 | 10 | 11 | 11 |
| 12 | 12 | 13 | 13 | 14 | 14 | 15 | 15 |

a) Calcule o tamanho total desta imagem **em bits**




# Image Compression

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 6 | 6 | 6 |
| 0 | 0 | 0 | 0 | 1 | 5 | 5 | 5 |
| 0 | 0 | 0 | 0 | 3 | 3 | 3 | 3 |
| 2 | 2 | 2 | 2 | 3 | 3 | 4 | 4 |
| 4 | 7 | 7 | 7 | 8 | 8 | 9 | 9 |
| 7 | 7 | 7 | 7 | 8 | 8 | 9 | 9 |
| 7 | 7 | 7 | 7 | 10 | 10 | 11 | 11 |
| 12 | 12 | 13 | 13 | 14 | 14 | 15 | 15 |

b) Aplique o RLE e calcule o tamanho (Suponha a imagem sendo um vetor obtido, juntando todas as linhas da matriz – de cima para baixo) e obtenha o tamanho **em bits**

**Use byte para as quantidades**
**Use Nyble para as tonalidades**

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 6 | 6 | 6 | 0 | 0 | 0 | 0 | 1 | 5 | 5 | 5 | ... |

***Nibble ou Nyble**