

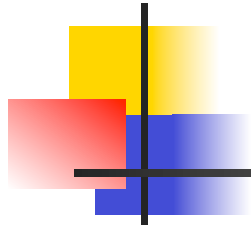
# SCC0561 – Multimídia

---

**Prof.: Dr. Marcelo G. Manzato**  
([mmanzato@icmc.usp.br](mailto:mmanzato@icmc.usp.br))

## Aula 4 – Compressão de Imagens - O Padrão JPEG.

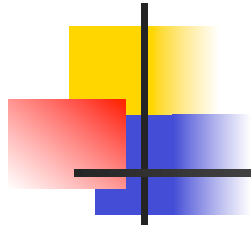
Instituto de Ciências Matemáticas e de Computação - ICMC  
Sala 3-111



# 1. O Padrão JPEG

---

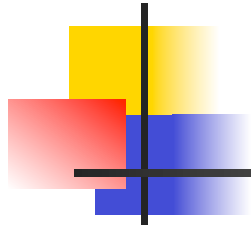
- O quê é JPEG?
- Preparação da imagem/bloco.
- Transformada DCT.
- Quantização.
- Codificação por Entropia.
- Construção do Quadro.



## 1.1 O quê é JPEG?

---

- Joint Photographic Experts Group.
  - ISO, CCITT e IEC.
  - Padrão para codificação de imagens estáticas de tons contínuos.
  - Possui 4 modos de operação:
    - **Seqüencial (*baseline mode*)**.
    - Progressivo.
    - Sem perdas.
    - Hierárquico.

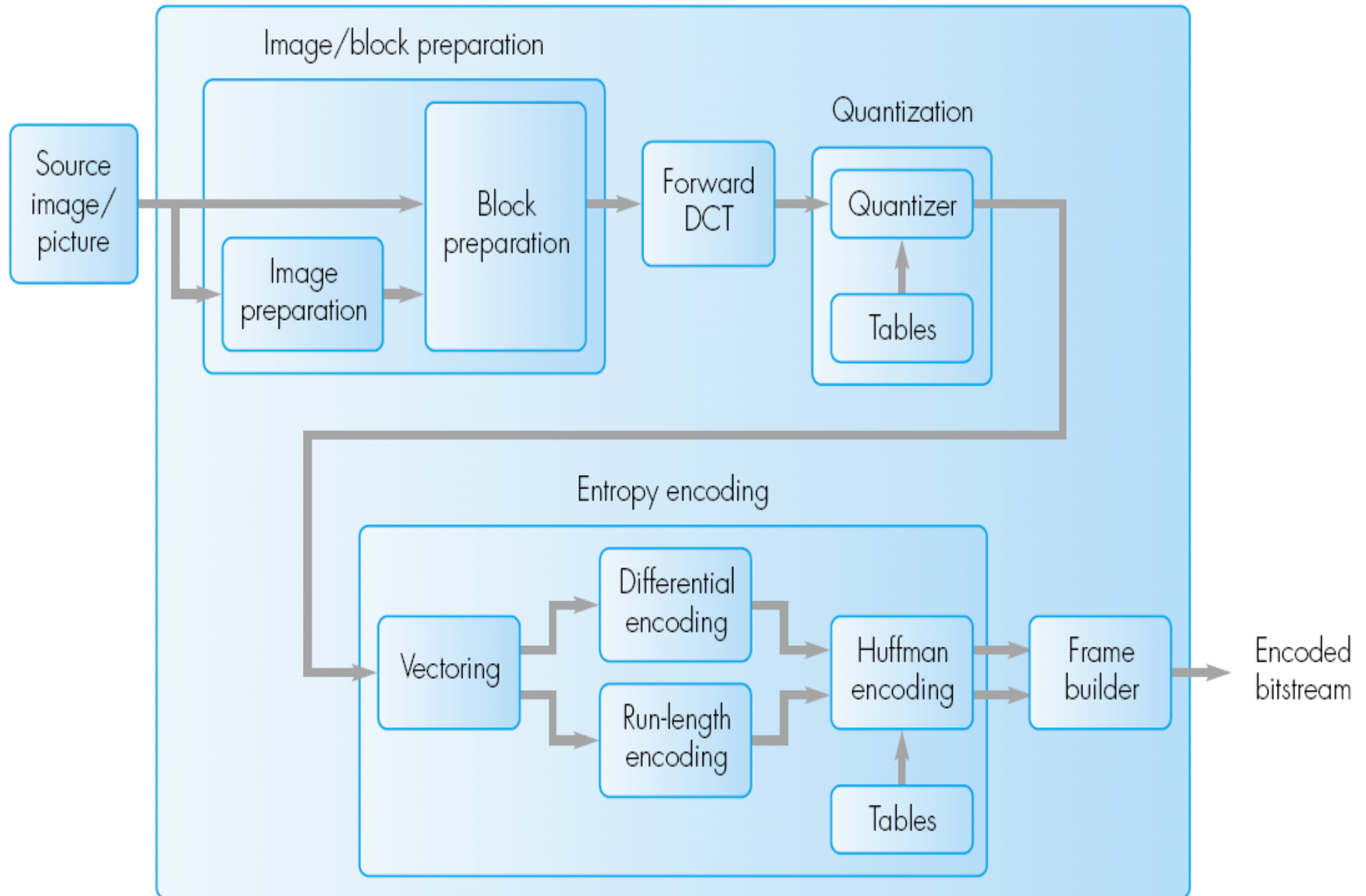


# 1.1 O quê é JPEG?

---

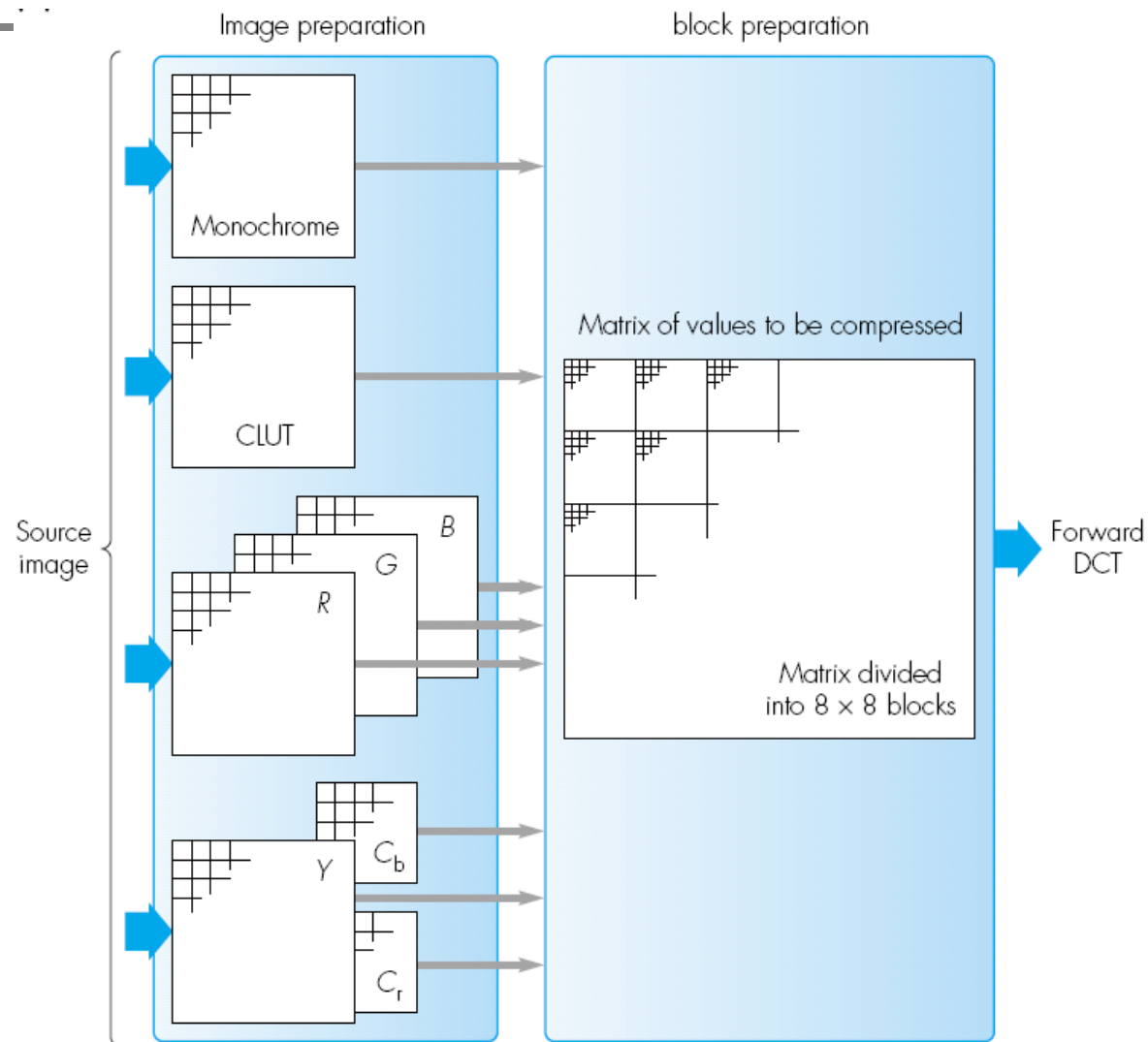
- Modo seqüencial
  - É um método de compressão com perdas.
  - Possui 5 etapas principais:
    - Preparação da imagem/bloco.
    - DCT.
    - Quantização.
    - Codificação.
    - Construção do quadro.

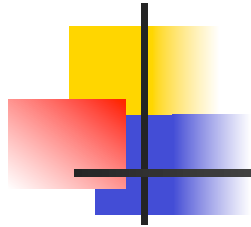
## JPEG encoder



## 1.2 Preparação da imagem/ bloco

- Imagem é dividida em blocos de 8 x 8 pixels.
- Isso permite aplicação mais eficiente da DCT.



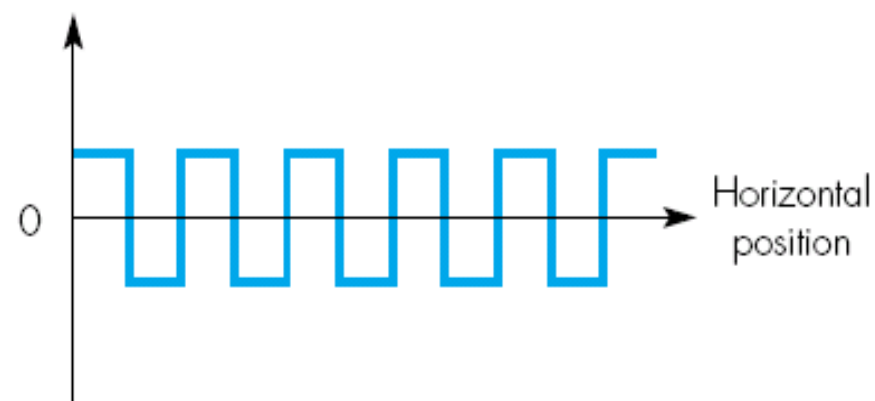
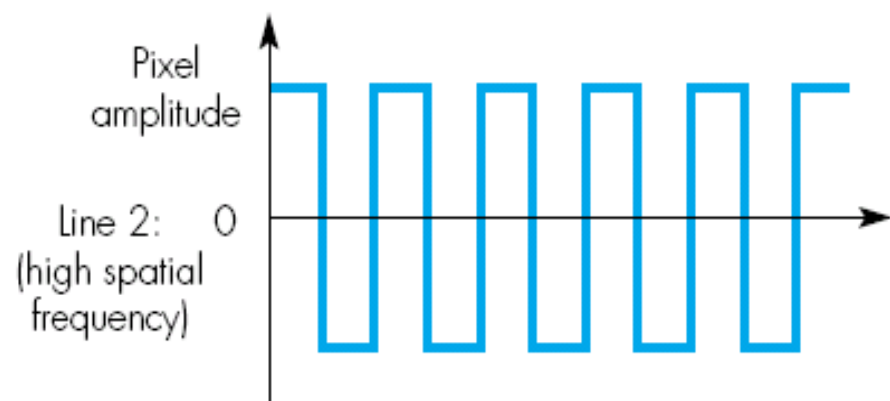
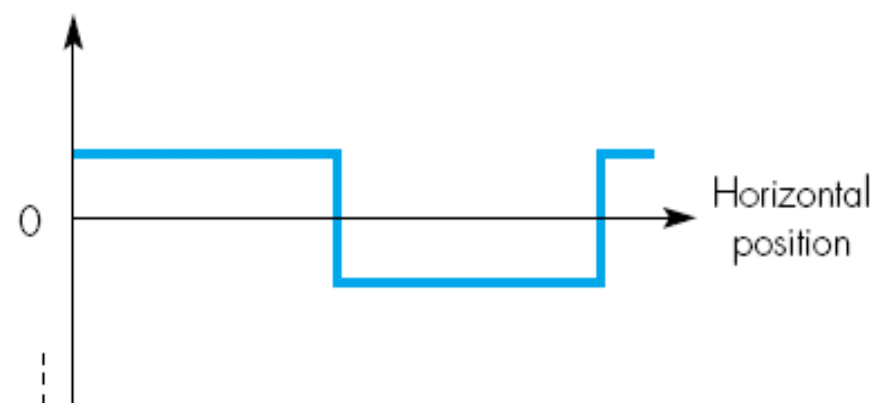
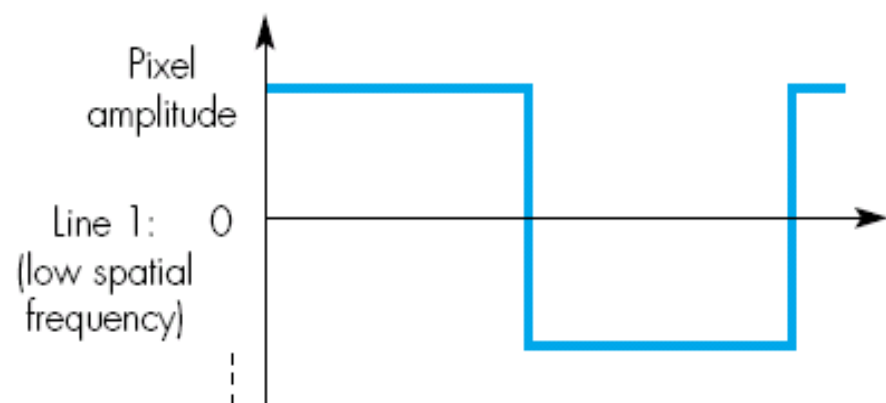
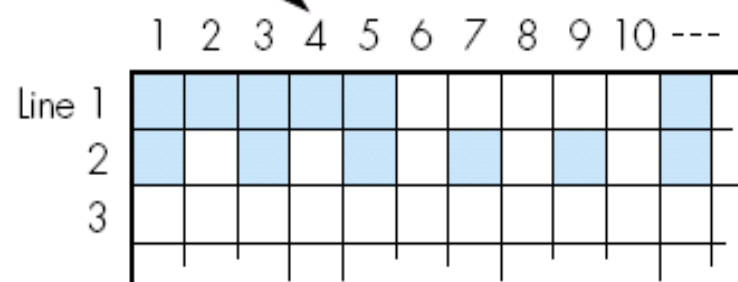
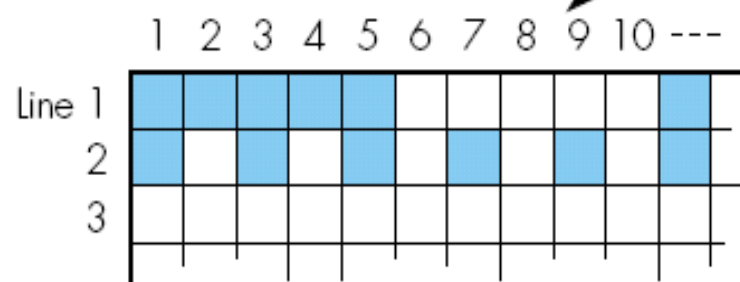


## 1.3 Transformada DCT

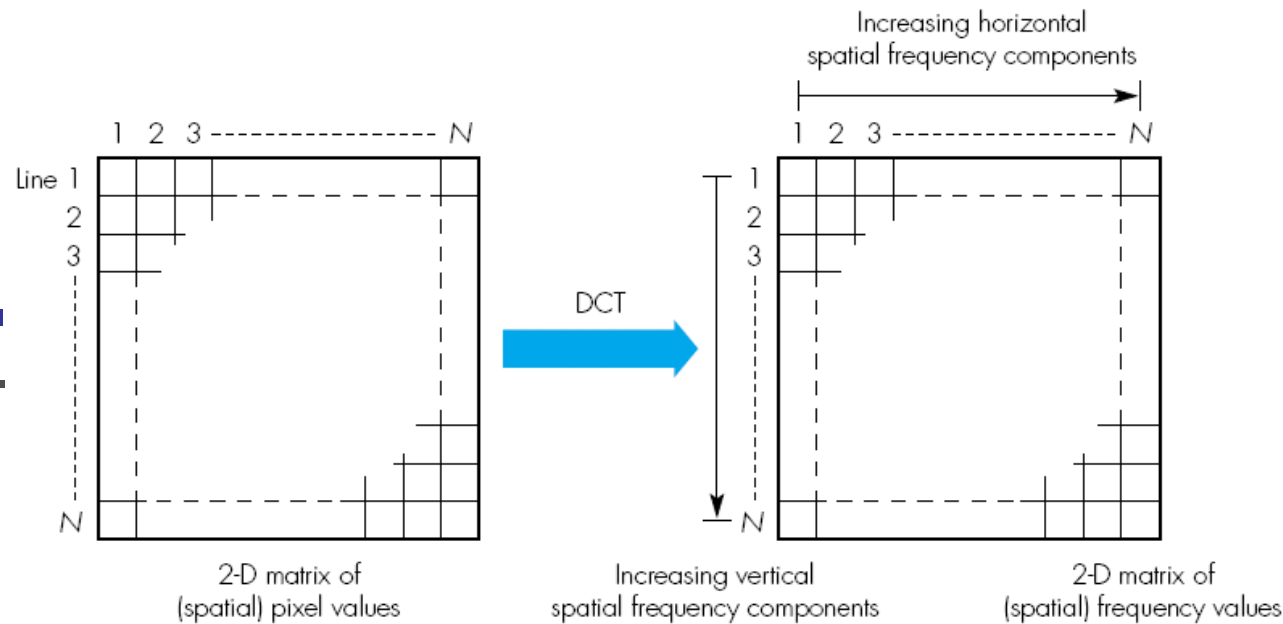
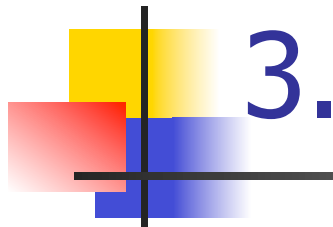
---

- Transformada Discreta de Cossenos (DCT).
- Transformadas:
  - Transformam a informação de um formato (domínio) para outro.
- Transformada DCT aplicada a imagens:
  - Transforma matriz (imagem) em matriz de frequências espaciais.
  - Não produz perdas.

Example pixel patterns







DCT = discrete cosine transform

$$F[i, j] = \frac{1}{4} C(i) C(j) \sum_{x=0}^7 \sum_{y=0}^7 P[x, y] \cos \frac{(2x+1)i\pi}{16} \cos \frac{(2y+1)j\pi}{16}$$

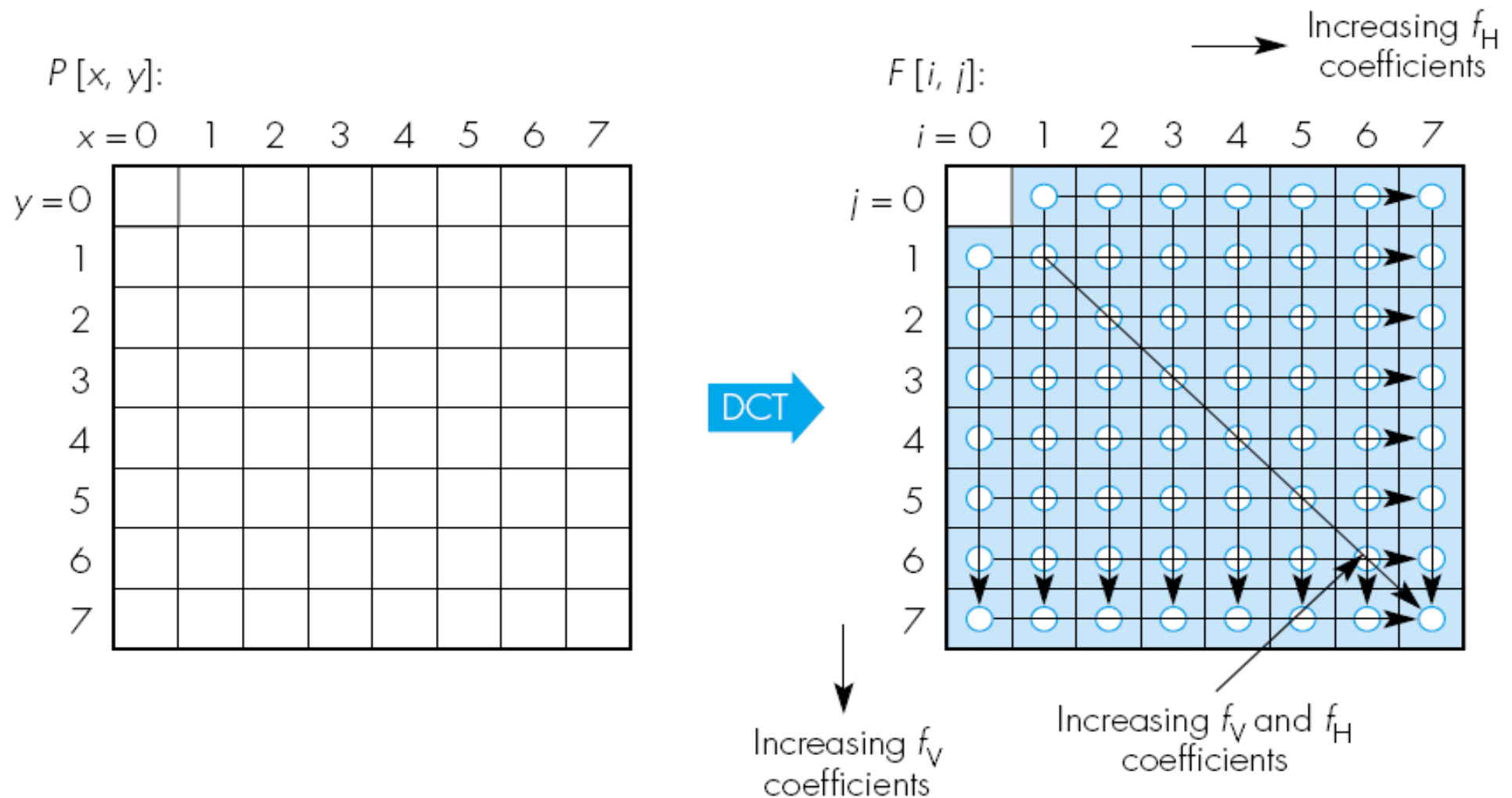
onde  $C(i)$  e  $C(j) = 1/\sqrt{2}$  para  $i, j=0$   
 $= 1$  para todos os outros valores de  $i$  e  $j$ .  
 $x, y, i$  e  $j$  todos variam de 0 a 7.

Transformada Discreta de Cossenos (DCT)

$$P[x, y] = \frac{1}{4} \sum_{i=0}^7 \sum_{j=0}^7 C(i) C(j) F[i, j] \cos \frac{(2x+1)i\pi}{16} \cos \frac{(2y+1)j\pi}{16}$$

onde  $C(i)$  e  $C(j) = 1/\sqrt{2}$  para  $i, j=0$   
 $= 1$  para todos os outros valores de  $i$  e  $j$ .  
 $x, y, i$  e  $j$  todos variam de 0 a 7.

Transformada Discreta de Cossenos Inversa (IDCT)



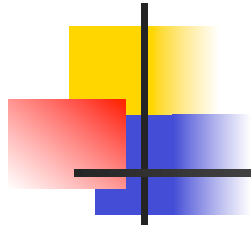
$P[x, y] = 8 \times 8$  matrix of pixel values

$F[i, j] = 8 \times 8$  matrix of transformed values/spatial frequency coefficients

In  $F[i, j]$ :  = DC coefficient     = AC coefficients

$f_H$  = horizontal spatial frequency coefficient

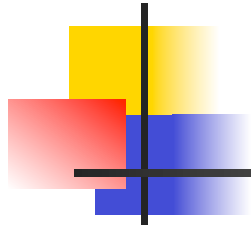
$f_V$  = vertical spatial frequency coefficient



## 1.3 Transformada DCT

---

- Olho humano é menos sensível a distorções em regiões com alta frequência espacial.
- Se a amplitude, nas altas frequências, está abaixo de um limite, o olho não detecta a informação.
- Matriz transformada ajuda a detectar e eliminar tais informações (redundância psicovisual).

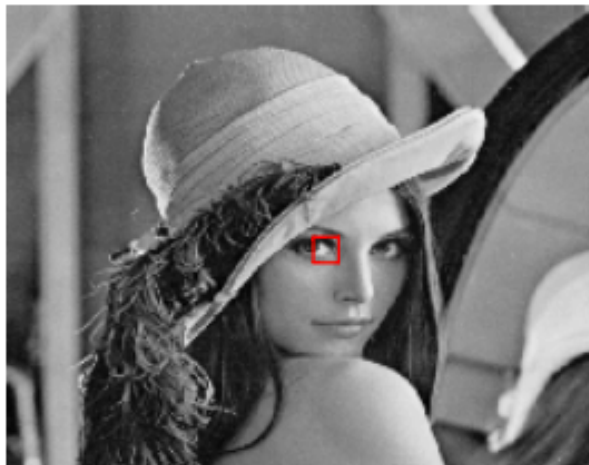


## 1.3 Transformada DCT

---

- Após DCT:
  - As regiões da imagem que possuem uma única cor geram matrizes com coeficientes DC idênticos (ou próximos) e poucos coeficientes AC.
  - As regiões da imagem que possuem transições de cores geram matrizes com coeficientes DC distintos e muitos coeficientes AC.
- Tamanho do bloco na imagem.
- Regiões com pouca/muita transição de cor X coeficientes DC/AC.

## 1.3 Transformada DCT



8x8



172	179	188	191	196	200	204	174
188	187	190	193	199	201	178	101
189	189	196	197	199	183	117	84
186	192	197	199	189	130	85	85
198	197	199	192	149	100	100	95
195	195	193	158	108	98	96	98
195	189	171	111	111	108	104	96
192	177	124	110	113	113	108	100

domínio espacial

DCT

1256,4	228,6	-50,0	17,7	-15,6	2	-2,7	5,8
154,8	-80	-93,2	27	-6,5	12,3	2	0,7
9,7	-92,3	57,3	39,3	-29	3,4	6,3	1,5
16,3	-12,7	35,4	-47,6	-6,9	17,8	-2,1	4,4
2,1	-18,2	4	-14,4	27,6	-5,7	-12,9	-1,4
-3	-3,9	0,6	-9,3	2,5	-17,8	12,3	6,1
-1,2	-5,4	1,9	-7,2	6,2	-1,5	6,2	-11,8
7,1	-2,9	3,8	0,9	-1,4	0	2	2,9

domínio de frequências



## 1.4 Quantização

---

- Quantização

- Olho humano:

- Boa resposta para coeficientes DC (baixa freq.).
    - Baixa resposta para coeficientes AC (alta freq.).

- Busca reduzir a quantidade de dados.

- Limite da amplitude para frequências: divide os valores da matriz transformada pelos valores correspondentes em uma tabela pré-definida.
  - Isso diminui os valores dos coeficientes proporcionalmente à posição dos mesmos na matriz.
  - Ocorre perda. No caso ideal, não perceptível.

DCT coefficients

120	100	90	80	60	40	32	10
90	88	80	72	58	40	28	8
80	76	70	66	44	38	20	6
60	58	54	52	44	26	16	5
50	46	42	40	26	14	12	5
30	28	25	23	10	8	6	3
15	11	10	8	7	6	4	1
5	4	4	3	3	2	1	1

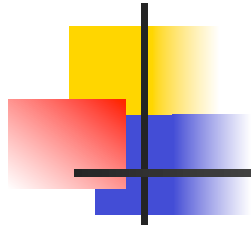
Quantized coefficients

12	10	6	4	3	2	1	0
9	6	4	3	2	2	0	0
6	4	3	2	1	1	0	0
3	3	2	2	1	1	0	0
2	2	2	1	1	0	0	0
1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Quantizer

10	10	15	20	25	30	35	40
10	15	20	25	30	35	40	50
15	20	25	30	35	40	50	60
20	25	30	35	40	50	60	70
25	30	35	40	50	60	70	80
30	35	40	50	60	70	80	90
35	40	50	60	70	80	90	100
40	50	60	70	80	90	100	110

Quantization table



## 1.4 Quantização

---

- Tabelas de quantização:
  - JPEG define duas tabelas *default*
    - Uma para luminância.
    - Uma para croma.
  - JPEG permite a utilização de tabelas personalizadas.





## 1.5 Codificação por Entropia

---

- Explora duas características da matriz quantizada:
  - Coeficiente DC será o maior valor da matriz
  - Muitos dos coeficientes de alta frequência serão zero



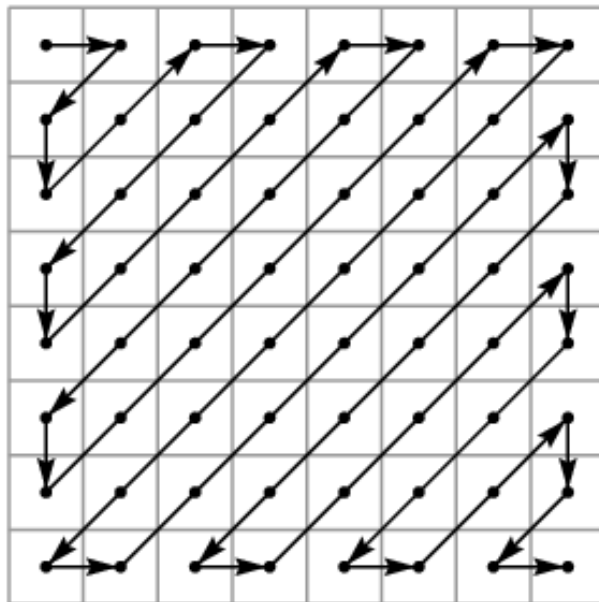
## 1.5 Codificação por Entropia

---

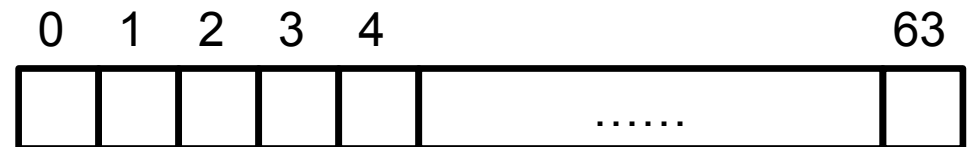
- Envolve quatro passos:
  - Vetorização.
  - Codificação por diferença.
  - Codificação por carreira (*run-length*).
  - Codificação Estatística (método de Huffman).

## 1.5 Codificação por Entropia

- Vetorização (zig-zag scan)

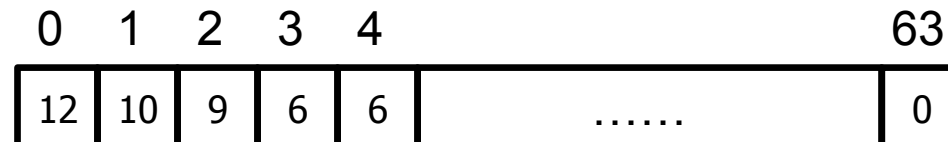


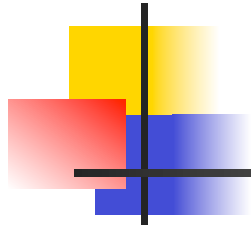
Vetor Linearizado



Coeficientes AC em ordem crescente por frequência

Coeficiente DC





## 1.5 Codificação por Entropia

---

- Codificação dos coeficientes DC
  - Codificação por diferença e Huffman.
  - DCs possuem alto grau de correlação (redundância espacial).
    - São blocos adjacentes na imagem.
  - Exemplo:
    - Seqüência de coeficientes DC de blocos adjacentes: 12, 13, 11, 11, 10, ...
    - Valores codificados: 12, 1, -2, 0, -1, ...



# 1.5 Codificação por Entropia

- Codificação dos coeficientes DC
  - Codificação na forma (SSS, value)
    - **SSS**: no. de bits necessários; **value**: bits

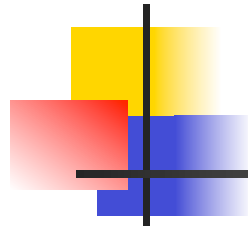
**Exemplo anterior:**

Seq. a ser codificada: 12, 1, -2, 0, -1

Valor	SSS	Value
12	4	1100
1	1	1
-2	2	01
0	0	
-1	1	0

(4,1100)(1,1)(2,01)(0)(1,0)

Difference value	Number of bits needed (SSS)	Encoded value
0	0	
-1, 1	1	1 = 1 , -1 = 0
-3, -2, 2, 3	2	2 = 10 , -2 = 01
		3 = 11 , -3 = 00
-7..-4, 4.. 7	3	4 = 100 , -4 = 011
		5 = 101 , -5 = 010
		6 = 110 , -6 = 001
		7 = 111 , -7 = 000
-15...-8, 8...15	4	8 = 1000 , -8 = 0111
		⋮



## 1.5 Codificação por Entropia

- Codificação dos coeficientes DC
  - **SSS**: codificados segundo uma árvore de Huffman pré-definida

### Exemplo anterior:

Seq. a ser codificada: 12, 1, -2, 0, -1

Valor	SSS	Value
12	4	1100
1	1	1
-2	2	01
0	0	010
-1	1	0

(101,1100)(011,1)(100,01)(010)(011,0)

Number of bits needed (SSS)	Huffman codeword
0	010
1	011
2	100
3	00
4	101
5	110
6	1110
7	11110
⋮	
11	111111110



## 1.5 Codificação por Entropia

---

- Codificação dos coeficientes AC
  - Codificação *run-length* (carreira).
  - Vetor de coeficientes possui longas cadeias de zeros.
  - Formato:  
(skip, value): **skip** indica a quantidade de zeros a ser “pulada”; **value** é o próximo valor não zero da seq.

18	5	8	7	0	1	0	0	4	3	9	8	2	0	1	0	.....	0
----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	-------	---



(0,5)(0,8)(0,7)(1,1)(2,4)(0,3)(0,9)(0,8)(0,2)(1,1)(0,0)



## 1.5 Codificação por Entropia

- Codificação dos coeficientes AC

(skip, value): **skip** indica a quantidade de zeros a ser “pulada”; **value** é o próximo valor não zero da seq.

- Campo **value** é também codificado na forma **SSS/value**
  - Ex.: (0,6) → Skip = 0; SSS = 3; Value=110

Difference value	Number of bits needed (SSS)	Encoded value
0	0	
-1, 1	1	1 = 1, -1 = 0
-3, -2, 2, 3	2	2 = 10, -2 = 01
		3 = 11, -3 = 00
-7..-4, 4..7	3	4 = 100, -4 = 011
		5 = 101, -5 = 010
		6 = 110, -6 = 001
		7 = 111, -7 = 000
-15...-8, 8...15	4	8 = 1000, -8 = 0111
		⋮



## 1.5 Codificação por Entropia

- Codificação dos coeficientes AC

(skip, value): **skip** indica a quantidade de zeros a ser “pulada”; **value** é o próximo valor não zero da seq.

- Campo **value** é também codificado na forma **SSS/value**

- Ex.: (0,6) → Skip = 0; SSS = 3; Value=110

codificado via  
árvore de Huffman

Difference value	Number of bits needed (SSS)	Encoded value
0	0	
-1, 1	1	1 = 1, -1 = 0
-3, -2, 2, 3	2	2 = 10, -2 = 01
		3 = 11, -3 = 00
-7..-4, 4..7	3	4 = 100, -4 = 011
		5 = 101, -5 = 010
		6 = 110, -6 = 001
		7 = 111, -7 = 000
-15...-8, 8...15	4	8 = 1000, -8 = 0111
		⋮



## 1.5 Codificação por Entropia

---

- Codificação estatística
  - Após a codificação Run-Length é aplicada uma codificação estatística.
    - JPEG usa Huffman.
  - A codificação estatística é aplicada no vetor inteiro, o que inclui o resultado das codificações dos DCs e ACs.
  - Vetor possui cadeias de bits – apropriado para codificação estatística.
  - JPEG usa tabela de códigos (prefixo).
    - São 256 códigos possíveis.
    - Pré-definida ou enviada junto com o *bitstream* da imagem.



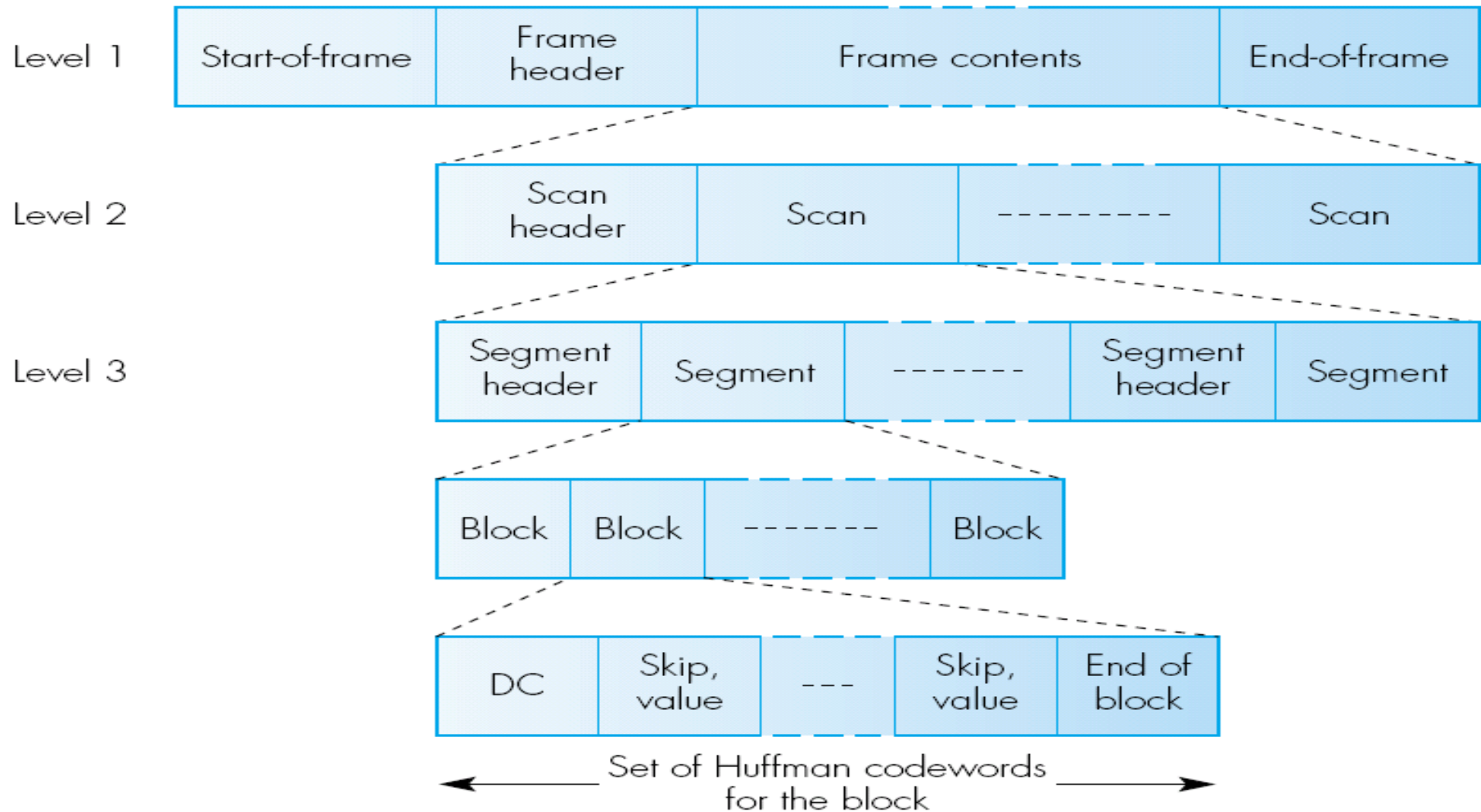
## 1.5 Codificação por Entropia

---

- Codificação estatística
  - Coeficientes DC
    - Codificação por Huffman nos bits do campo SSS
  - Coeficientes AC
    - Bits em Skip e SSS são tratados como um único símbolo e codificados segundo tabela Huffman contendo todas as combinações possíveis

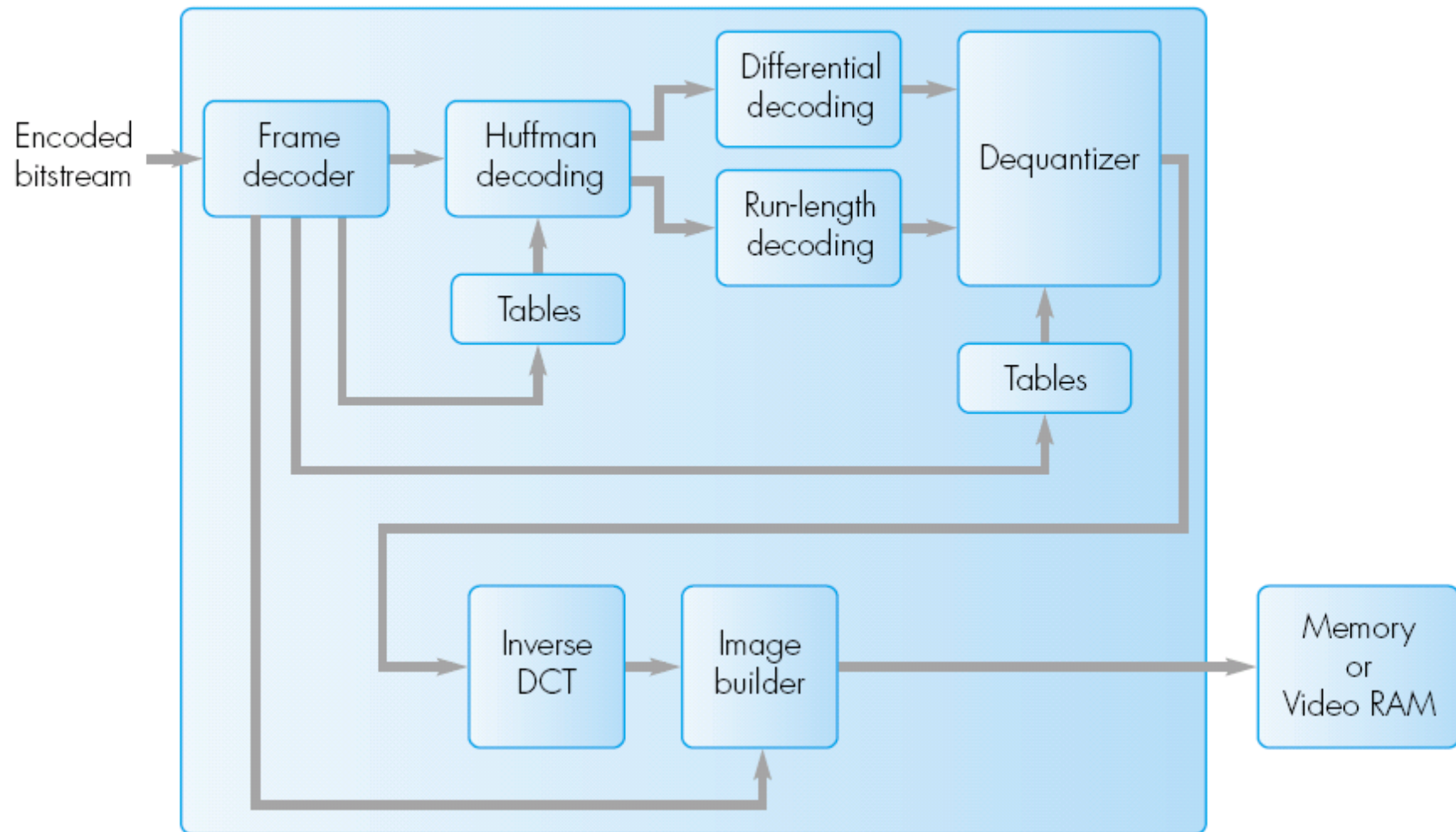


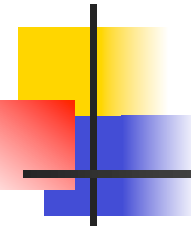
## 1.6 Construção do Quadro



## 1.7 Decodificação

JPEG decoder





## Exemplo – bloco original

140	144	147	140	139	155	179	175
144	152	140	147	140	148	167	179
152	155	136	167	163	162	152	172
168	145	156	160	152	155	136	160
162	148	156	148	140	136	147	162
147	167	140	155	155	140	136	162
136	156	123	167	162	144	140	147
148	155	136	155	152	147	147	136



## Bloco com shifting

---

12	16	19	12	11	27	51	47
16	24	12	19	12	20	39	51
24	27	8	39	35	34	24	44
40	17	28	32	24	27	8	32
34	20	28	20	12	8	19	34
19	39	12	27	27	12	8	34
8	28	-5	39	34	16	12	19
20	27	8	27	24	19	19	8

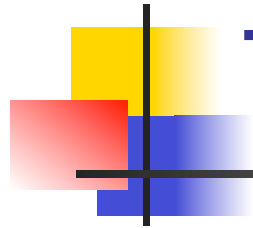


# Após a DCT

---

$$\begin{bmatrix} 185.88 & -17.962 & 14.943 & -9.0778 & 23.125 & -9.0856 & -13.901 & -19.110 \\ 20.365 & -34.045 & 26.557 & -9.1747 & -11.106 & 10.935 & 13.866 & 6.7143 \\ -10.547 & -23.469 & -1.6402 & 5.9121 & -18.238 & 3.3890 & -20.329 & -1.0530 \\ -8.2518 & -5.0009 & 14.524 & -14.729 & -8.3648 & -2.5596 & -3.0050 & 8.2253 \\ -3.3750 & 9.5359 & 8.0480 & 1.2188 & -11.125 & 18.051 & 18.450 & 15.068 \\ 3.7574 & -2.1876 & -18.039 & 8.4227 & 8.1706 & -3.4929 & 0.92215 & -6.9987 \\ 8.8337 & 0.65168 & -2.8289 & 3.5882 & -1.2401 & -7.3423 & -1.1098 & -2.0184 \\ 0.014635 & -7.8035 & -2.3794 & 1.5633 & 1.1648 & 4.2876 & -6.3987 & 0.26693 \end{bmatrix}$$





# Tabela de quantização

---

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31



# Após quantização

---

$$\begin{bmatrix} 62 & -4 & 2 & -1 & 2 & -1 & -1 & -1 \\ 4 & -5 & 3 & -1 & -1 & 1 & 1 & 0 \\ -2 & -3 & 0 & 0 & -1 & 0 & -1 & 0 \\ -1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & -1 & 1 & 1 & 1 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



# Últimos passos

---

- **Zig-zag sequence**

- 62, -4, 4, -2, -5, 2, -1, 3, -3, -1, 0, 0, 0, -1, 2, -1, -1, 0, 1, 1, 0, 1, 0, 1, -1, -1, 1, -1, -1, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0, -1, 0, -1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0

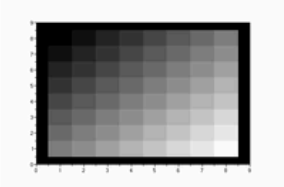
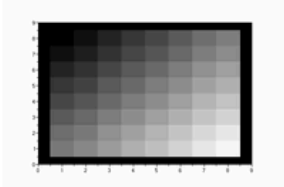
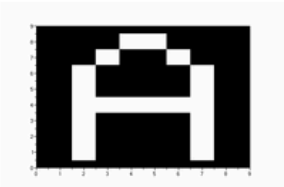
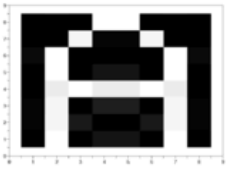
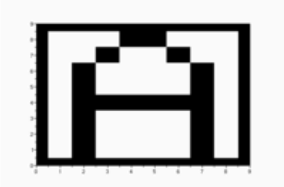
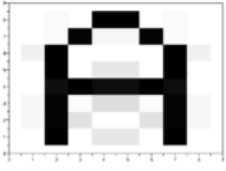
- **Intermediate symbol sequence**

- (6)(62), (0,3)(-4), (0,3)(4), (0,2)(-2), (0,3)(-5), (0,2)(2), (0,1)(-1), (0,2)(3), (0,2)(-3), (0,1)(-1), (3,1)(-1), (0,2)(2), (0,1)(-1), (0,1)(-1), (1,1)(1), (0,1)(1), (1,1)(1), (1,1)(1), (0,1)(-1), (0,1)(-1), (0,1)(1), (0,1)(-1), (0,1)(-1), (0,1)(1), (3,1)(-1), (5,1)(-1), (1,1)(-1), (3,1)(1), (6,1)(1), (1,1)(1), (0,0)

- **Encoded bit sequence (total 154 bits)**

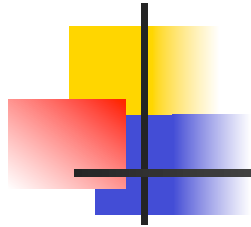
- (1110)(111110) (100)(011) (100)(100) (01)(01) (100)(010) (01)(10) (00)(0) (01)(11) (01)(00) (00)(0) (111010)(0) (01)(10) (00)(0) (00)(0) (1100)(1) (00)(1) (1100)(1) (1100)(1) (00)(0) (00)(0) (00)(1) (00)(0) (00)(0) (00)(0) (00)(1) (111010)(0) (1111010)(0) (1100)(0) (111010)(1) (1111011)(1) (1100)(1) (1010)

## 1.8 Considerações Sobre JPEG

	
Degradê de cinzas sem passar por DCT	Degradê de cinzas após passar por DCT
	
Letra A com fundo preto sem passar por DCT	Letra A com fundo preto após passar por DCT
	
Letra A com fundo branco sem passar por DCT	Letra A com fundo branco após passar por DCT

Exemplo de imagens antes e após DCT + quantização.

Melhor desempenho em imagens com transição suave de cores



## 1.8 Considerações Sobre JPEG

---

- Padrão abrangente.
- Alcança boas taxas de compressão para imagens de tons contínuos. (até 20:1).
- Desempenho diminui em imagens com muita transição de cores.
- Baseado em particularidades do sistema visual humano:
  - Não é necessário reproduzir cantos com fidelidade.
  - O olho humano não responde bem a transições nas altas frequências espaciais.
  - É adequado para imagens de tom contínuo.



# Para Saber Mais

---

- Gonzales & Woods. Digital Image Processing. 2nd ed. Prentice-Hall, 2002. Capítulo 8, seção 8.1.
- Halsall, F. Multimedia Communications: Applications, Networks, Protocols, and Standards, Addison-Wesley Publishing, 2001. ISBN: 0201398184. Capítulo 2, seção 2.4 e capítulo 3, seções 3.2 e 3.4.
- Pennebaker & Mitchell. JPEG Still Image Data Compression Standard. Van Nostrand Reinhold, 1993.