UNIVERSIDADE FEDERAL DE PELOTAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

TEC2/TEC4: REDES MULTIMÍDIA (RMM)

Unidade 3
Codificação de Imagem

Prof. Guilherme Corrêa
gcorrea@inf.ufpel.edu.br

Sumário

- Introdução
- Formato JPEG
- Formato BMP
- Formato GIF
- Formato TIFF
- Formato PNG

Sumário

- **Introdução**
- Formato JPEG
- Formato BMP
- Formato GIF
- Formato TIFF
- Formato PNG

Introdução

- Requisitos de largura de banda para imagem:
  - 640x480 pixels/imagem e *true color* (3 bytes)
    - 640x480x3 = 921,6 KB
    - 115 segundos para transferir em um enlace de 64 Kbps!
- Muitos formatos
  - JPEG
  - BMP (*Bitmap Format*)
  - GIF (*Graphics Interchange Format*)
  - TIFF (*Tagged Image File Format*)
  - PNG (*Portable Network Graphics*)

Sumário

- Introdução
- **Formato JPEG**
- Formato BMP
- Formato GIF
- Formato TIFF
- Formato PNG

Formato JPEG

- *Joint Photographic Experts Group* (JPEG)
- Tornou-se um padrão em 1992
  - Padrão ISO: IS 10918
- Tira proveito das **limitações do sistema visual humano**
  - Cérebro/olhos não conseguem perceber detalhes extremamente finos
  - Especialmente informação de cor
- Taxa de compressão: 3:1 até 100:1
  - Compressão típica 10:1 a 20:1
- Compressão com perdas, mas quase imperceptíveis!

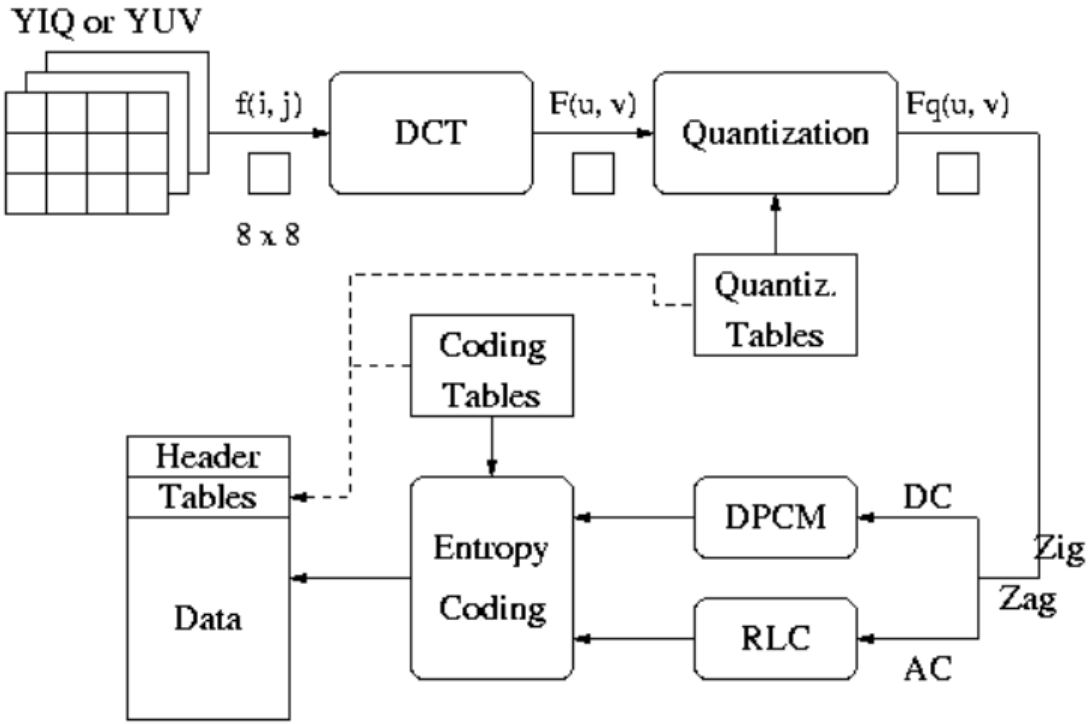
Formato JPEG

476x327x24 467KB

JPEG: 45KB (1:10.4)

JPEG: 11.6KB (1:40)

Codificação JPEG



Codificação JPEG

- Imagens monocromáticas/tons de cinza
  - Uma matriz
- Espaço de cores RGB
  - Três matrizes (verde, verde, azul)
- Espaço de cores **YUV**
  - Três matrizes
    - luminância (**Y**)
    - crominância azul (**Cb**)
    - crominancia vermelha (**Cr**)
  - Matrizes Cb e Cr podem ser menores
- Cada elemento da matriz pode ter 8 ou 12 bits

Codificação JPEG

Espaço de cores YUV

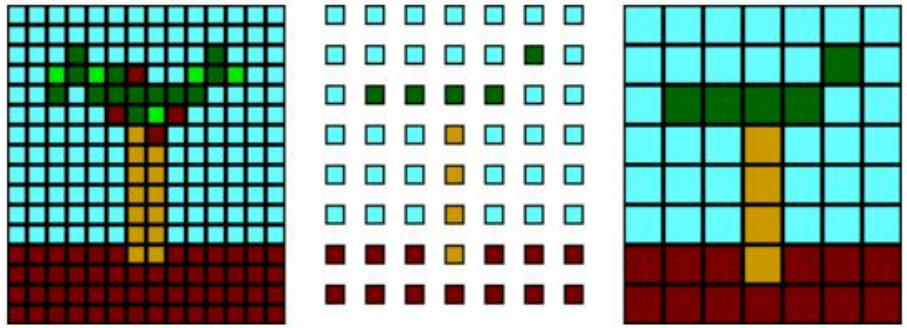
- Sistema visual humano é menos sensível à variação de cores do que de iluminação
- Subamostragem de cores
  - 1 byte para cada componente de luminância
  - 4 bits para cada componente de crominância (codificação "4:2:2")
  - Usa 2/3 do espaço usado por RGB

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Codificação JPEG

Subamostragem

- 4:4:4: sem subamostragem, imagem original
- 4:2:2: uma coluna sim (armazenada), outra não (descartada)
- 4:2:0: uma coluna sim, outra não; uma linha sim, outra não
- Na hora de mostrar, o tamanho do pixel é duplicado (*pixel doubling*)



Discrete Cosine Transform

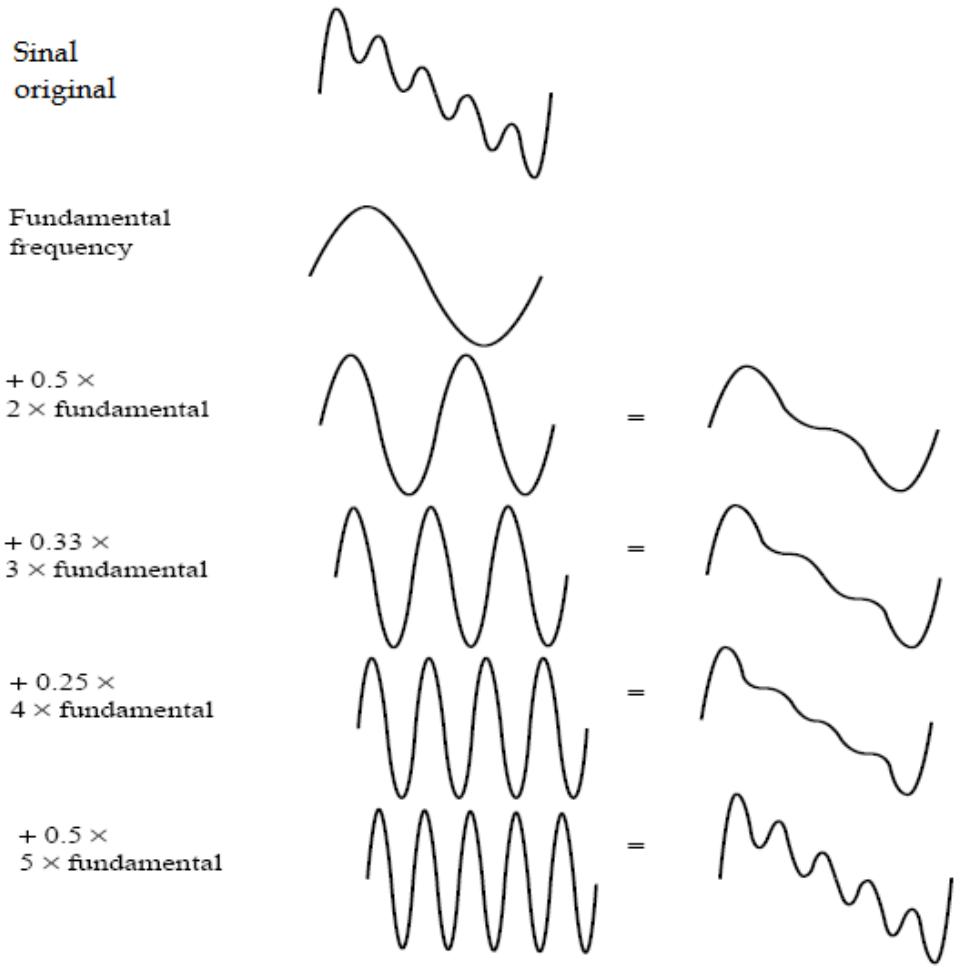
- Transforma informação do domínio espacial para o domínio das frequências
- *Entrada*: bloco de pixels
- *Saída*: coeficientes DCT (frequências espaciais)
  - Correspondem ao quanto os valores dos pixels variam, em função da sua posição dentro do bloco
  - Muitas variações: imagem com muito detalhe fino
  - Poucas variações: mudança uniforme de cor e pouco detalhe fino
- Quando há pouca variação nos pixels: apenas alguns coeficientes são suficientes para representar a imagem
- **DCT não comprime!** Apenas reorganiza os dados para que a próxima etapa possa comprimir mais eficientemente.

DCT de 1 dimensão

- Imagem consistindo em um array de valores de pixels, que variam no espaço, pode ser representada como a soma de *N* componentes de frequência
- Relembrando a série de Fourier...

Relembrando a Série de Fourier

Série de Fourier



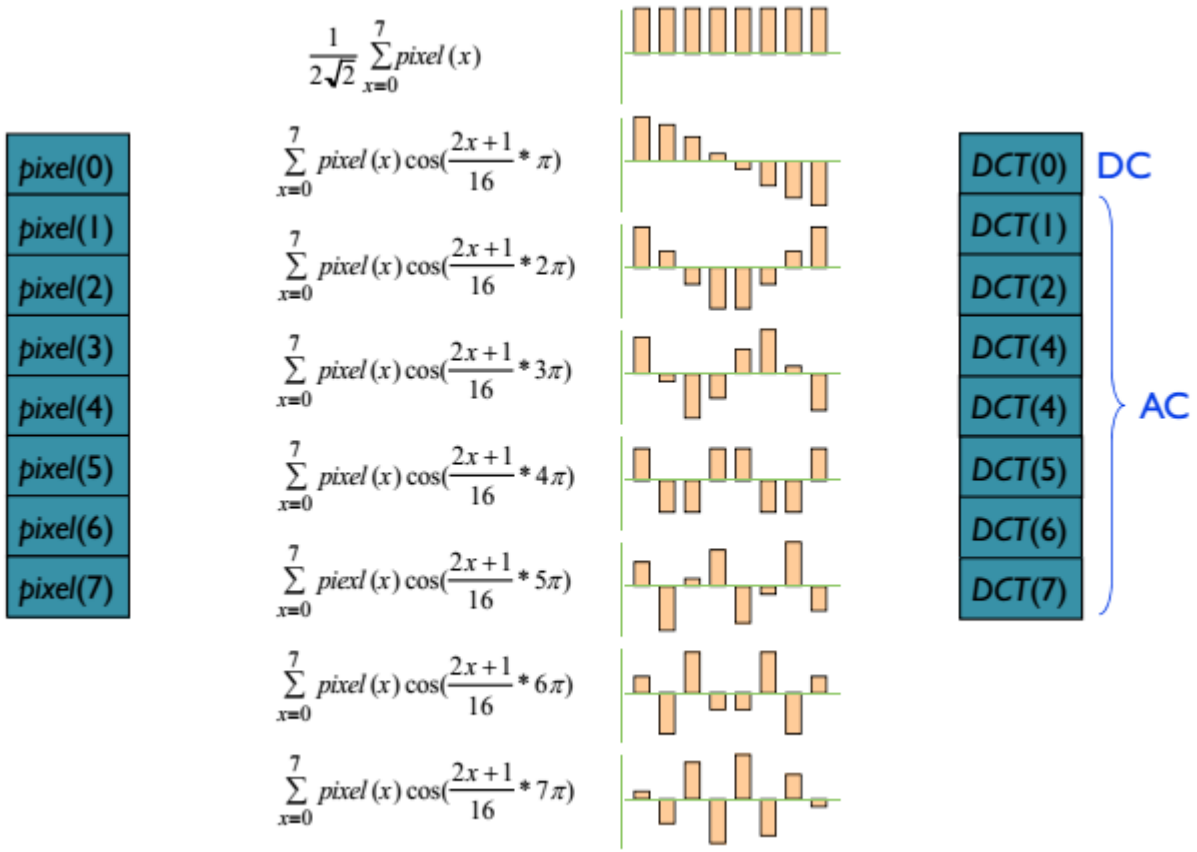
DCT de 1 dimensão

- Imagem consistindo em um array de valores de pixels, que variam no espaço, pode ser representada como a soma de *N* componentes de frequência
- Tipicamente, as mudanças em intensidade são muito pequenas, com poucas bordas acentuadas
  - Pouca ou nenhuma contribuição dos componentes de alta frequência!
- Dois tipos de componentes:
  - DC: único, relacionado à média dos valores de pixels em um bloco
  - AC: representam as frequências de variações de pixels no bloco

$$DCT(i) = \sqrt{\frac{2}{N}} C(i) \sum_{x=0}^{N-1} pixel(x) \cos \frac{(2x+1)i\pi}{2N}$$

$$C(i) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } i = 0 \text{ (DC)} \\ 1 & \text{if } i > 0 \text{ (AC)} \end{cases}$$

DCT de 1 dimensão



DCT de 2 dimensões

- Componente chave no JPEG e no MPEG!
- Soma ponderada de funções cosseno 8x8

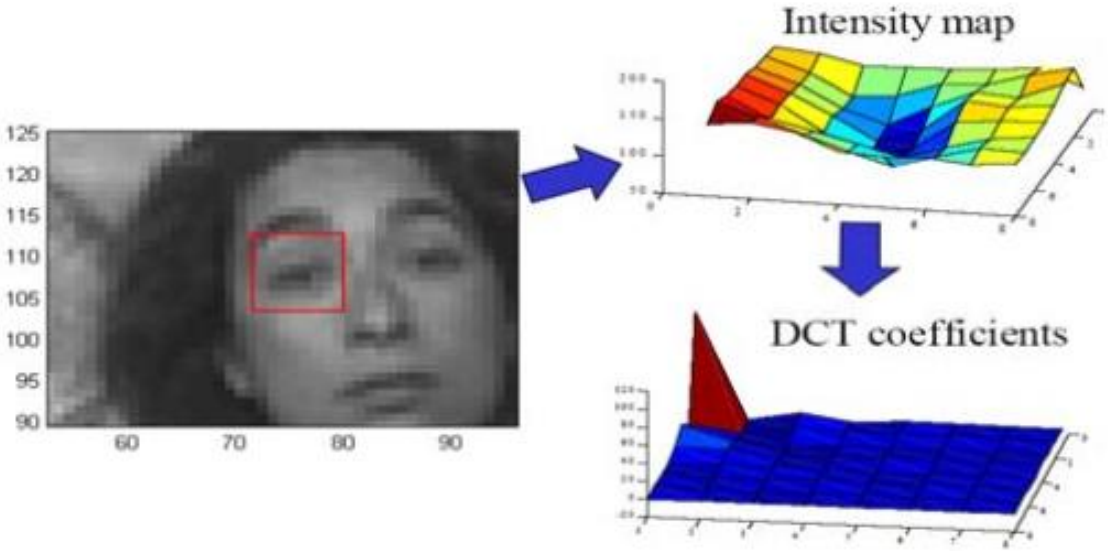
$$DCT(i,j) = \frac{1}{4} C(i)C(j) \sum_{x=0}^{7} \sum_{y=0}^{7} pixel(x,y) \cos\frac{(2x+1)i\pi}{16} \cos\frac{(2y+1)j\pi}{16}$$

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

$$C(x) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases}$$

DCT de 2 dimensões

- Componente chave no JPEG e no MPEG!
- Soma ponderada de funções cosseno 8x8



Exemplo: DCT no JPEG

- Divide imagem em blocos 8x8
- Valores dos pixels são deslocados para a faixa [-128, 127], com zero no centro
- Pixels de 8 bits produzem coeficientes DCT de 12 bits (com sinal)
  - Alguns erros de arredondamento, mas mínimos

Exemplo: DCT no JPEG

Resolution 720x572 pixels

DC

40	38	45	40	43	54	60	58
39	36	44	32	47	69	77	85
50	40	25	54	66	60	33	32
57	38	38	66	47	11	2	5
59	36	47	62	24	2	9	11
58	41	55	53	6	4	10	1
58	33	57	39	3	5	4	2
64	44	54	35	3	7	3	3

Block at 8x8 pixels

Color value matrix

44	6	0	-4	0	-1	0	0
12	0	-3	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	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

DCT coefficients

Baixa frequência (AC)

Baixa frequência (AC)

Quantização

- Tenta determinar quais informações podem ser descartadas com segurança, sem perda significativa de qualidade visual
  - Sistema visual menos sensível à crominância que luminância
  - Não percebemos mudanças sutis de cor (isto é, componentes AC pequenos)
  - Percebemos melhor mudanças mais acentuadas de cor (isto é, componentes AC de baixa frequência)
- Reduzir precisão → reduzir o número de bits
- Dividir coeficientes DCT(i,j) por coeficiente de quantização Q(i,j) e truncar

$$DCT_Q(i,j) = \left\lfloor \frac{DCT(i,j)}{Q(i,j)} \right\rfloor$$

Quantização

- Usa tabelas de quantização derivadas a partir de medições empíricas extensas

Luminance

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

Chrominance

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

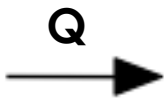
- Muitos componentes de alta frequência acabam sendo zerados: alta taxa de compressão!

Quantização

Exemplo

Saída da Transformada

-784	0	-164	0	-16	0	-19	0
0	0	0	0	0	0	0	0
-164	0	137	0	-21	0	11	0
0	0	0	0	0	0	0	0
-16	0	-21	0	48	0	-9	0
0	0	0	0	0	0	0	0
-19	0	11	0	-9	0	23	0
0	0	0	0	0	0	0	0

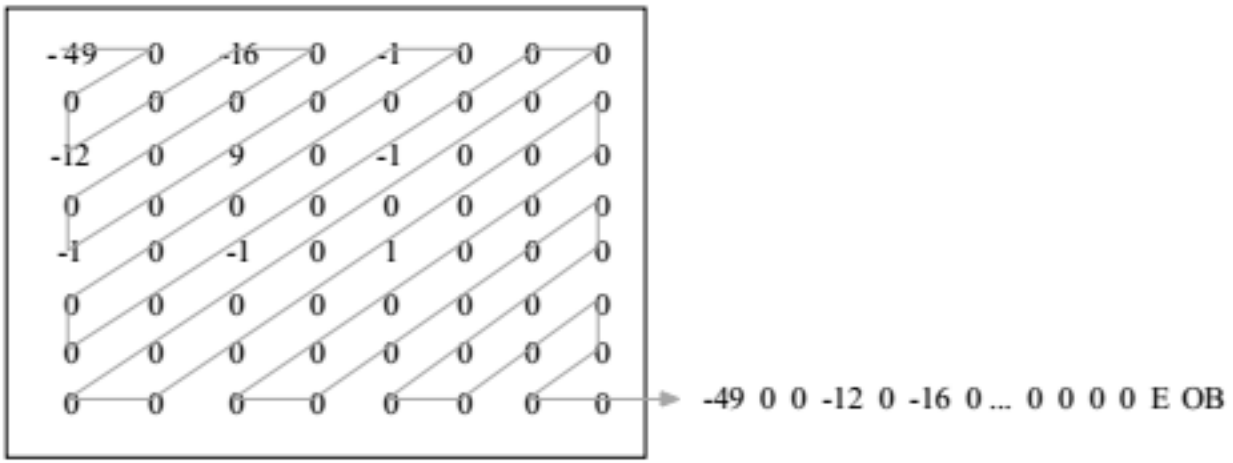


Saída da Quantização

-49	0	-16	0	-1	0	0	0
0	0	0	0	0	0	0	0
-12	0	9	0	-1	0	0	0
0	0	0	0	0	0	0	0
-1	0	-1	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

Scan em Zig-Zag

- O comprimento da sequência de valores zero pode ser aumentado se os componentes de baixa frequência forem colocados antes dos componentes de alta frequência



Compressão sem Perdas

- Aqui é onde entra a compressão de verdade!
- Componentes DC e AC são tratados de forma diferente
  - Coeficientes DC determinam a cor básica (ou intensidade luminosa) de um bloco

Codificação do Coeficiente DC

- **Codificação DC diferencial**
- Gerada através do cálculo de DIFF(*x*) = DC(*x*) - DC(*x*-1)
  - *x* e *x*-1 representam dois bloco 8x8 sucessivos
  - Tipicamente há muita correlação entre os DCs de dois blocos 8x8 adjacentes
  - DIFF tende a ser pequeno!

- Codificação no formato (*s1*, *s2*), onde:
  - *s1*: número de bits necessários para codificar o valor, codificado com Huffman (*Variable Length Coding* – VLC)
  - *s2*: código que representa DIFF, codificado com *Variable Length Integer* – VLI.

Codificação do Coeficiente DC

VLC Coding of Symbol-1

Bit Size(m)	Huffman Code	Differential DC Coeff.Value
0	00	0
1	010	(-1,1)
2	011	(-3,-2)(2,3)
3	100	(-7…-4)(4…7)
4	101	(-15…-8)(8…15)
5	110	(-31…-16)(16…31)
6	1110	(-63…-31)(32…63)
7	11110	(-127…-64)(64…127)
8	111110	(255…-128)(128…255)
9	1111110	(-511…-256)(256…511)
10	11111110	(-1023…-512)(512…511)
11	111111110	(-2047…-1024)(1024…2047)

Codificação do Coeficiente DC

Codificação VLC do símbolo 1 (*s1*):

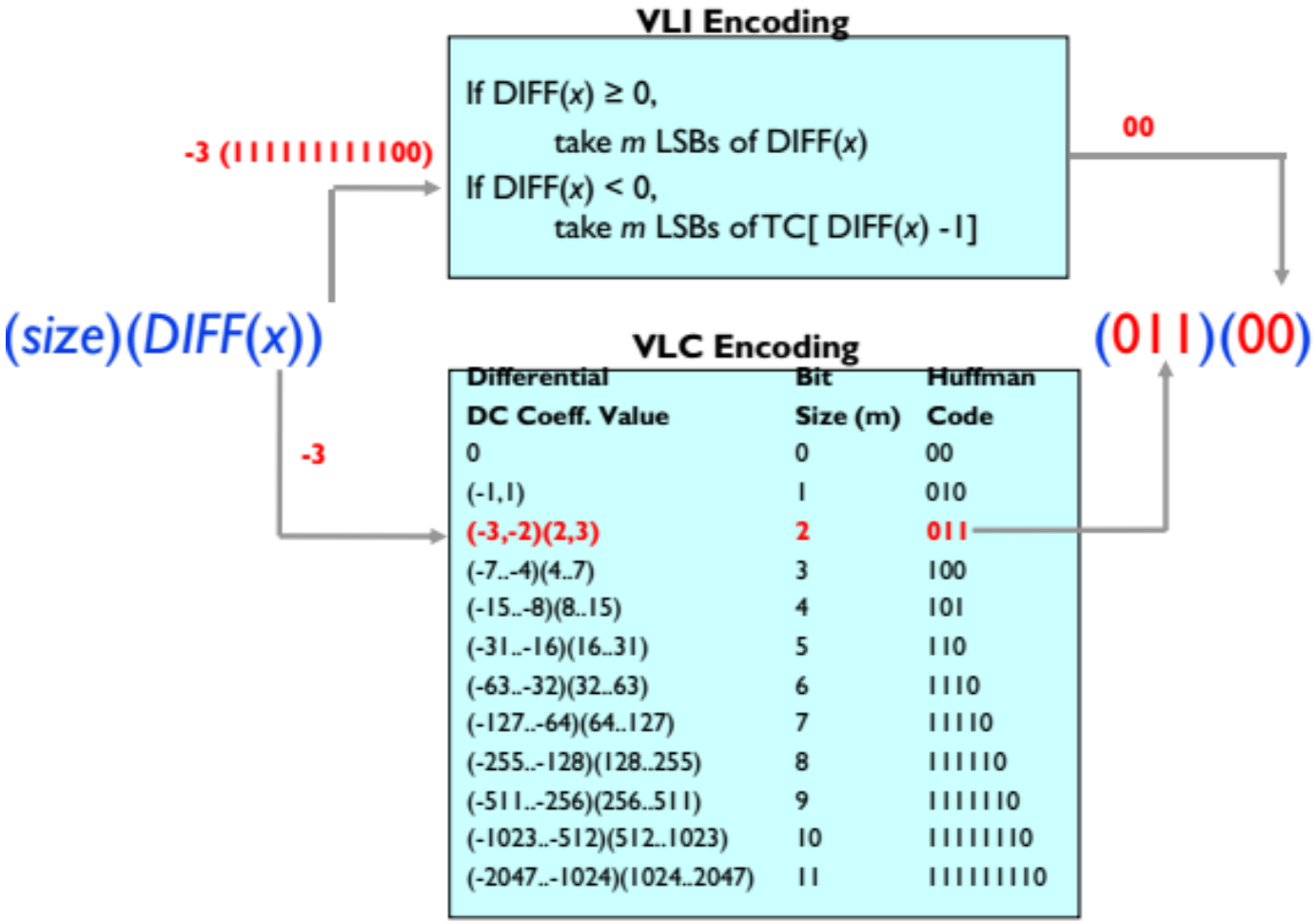
- Ex.: DIFF(x) = 3 → olhar a tabela de Huffman → VLC = 011

Codificação VLC do símbolo 2 (*s2*):

- Se DIFF(x) ≥ 0, usa os *m* LSBs de DIFF(x)
  - Ex.: DIFF(x) = 3 → 00000000011 → VLI = 11
- Se DIFF(x) < 0, usa os *m* LSBs do complemento de 2 de DIFF(x)-1
  - Ex.: DIFF(x) = -3 → C2[DIFF(x)-1] = 11111111100 → VLI = 00

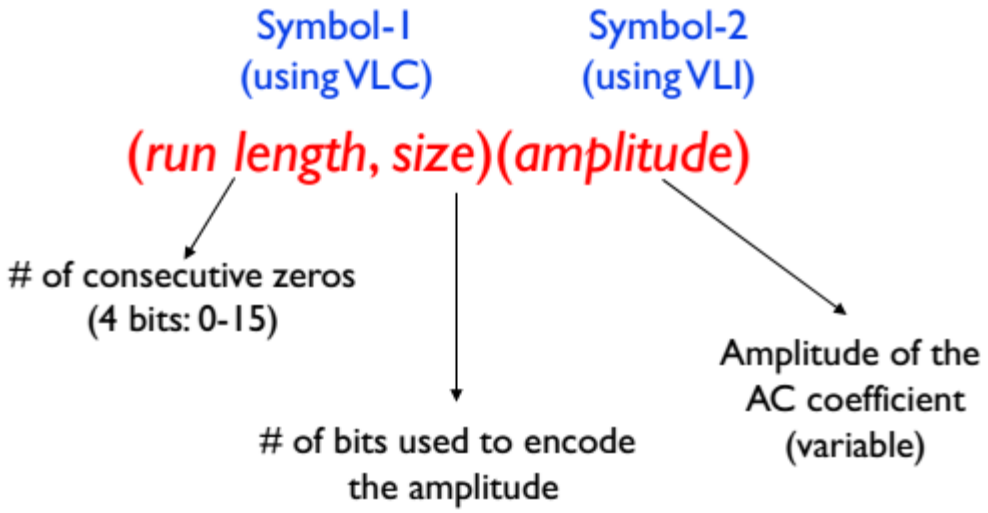
LSB = *Least Significant Bits* (bits menos significativos)
C2 = complemento de dois

Codificação do Coeficiente DC



Codificação do Coeficiente AC

- Codificação similar à do coeficiente DC, mas usa RLE
  - RLE: *Run Length Encoding*
  - Se aproveita da tendência de existirem muitos zeros

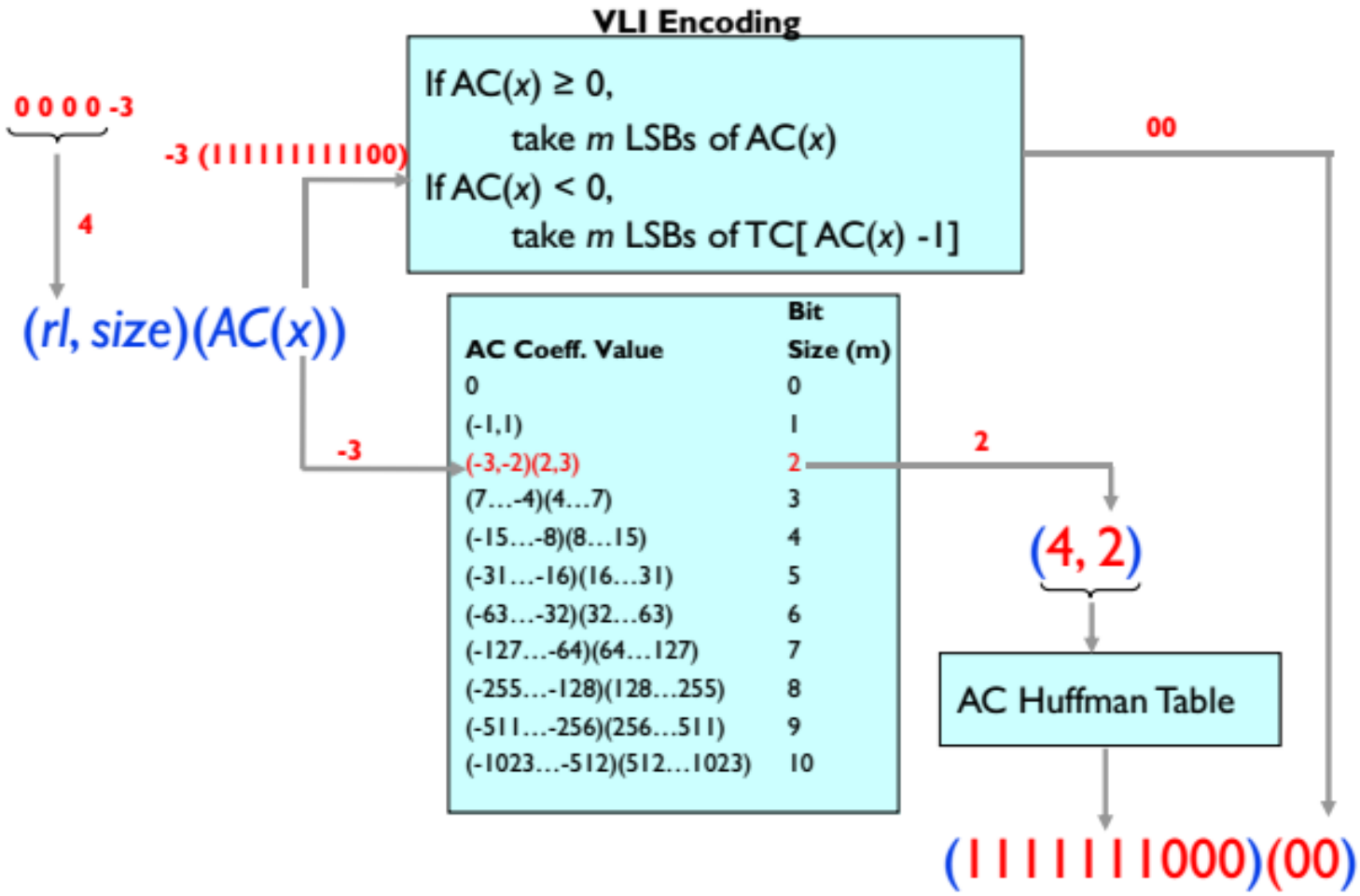


Codificação do Coeficiente AC

- Codificação similar à do coeficiente DC, mas usa RLE
  - RLE: *Run Length Encoding*
  - Se aproveita da tendência de existirem muitos zeros

Bit Size (m)	AC Coeff. Value
0	0
1	(-1,1)
2	(-3,-2)(2,3)
3	(-7..-4)(4..7)
4	(-15..-8)(8..15)
5	(-31..-16)(16..31)
6	(-63..-32)(32..63)
7	(-127..-64)(64..127)
8	(-255..-128)(128..255)
9	(-511..-256)(256..511)
10	(-1023..-512)(512..1023)

Codificação do Coeficiente AC



Exemplo de Codificação

Original zig-zag sequence

(-49,)(0, 0, -12,) (0, -16,) (0, 0, 0, 0, -1,)(0, 9,)(0, -1,)(0, 0, 0, 0, 0, 0, 0, 0, -1,)(0, -1,)(0, 0, 0, 0, 0, 0, 0, 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, 0, 0, 0, 0, 0, 0, 0, EOB)

Decimal equivalent

(6)(-49) (2,4)(-12) (1,5)(-16) (4, 1)(-1) (1,4)(9) (1,1)(-1)
(8,1)(-1) (1,1)(-1) (13,1)(1) (0,0)

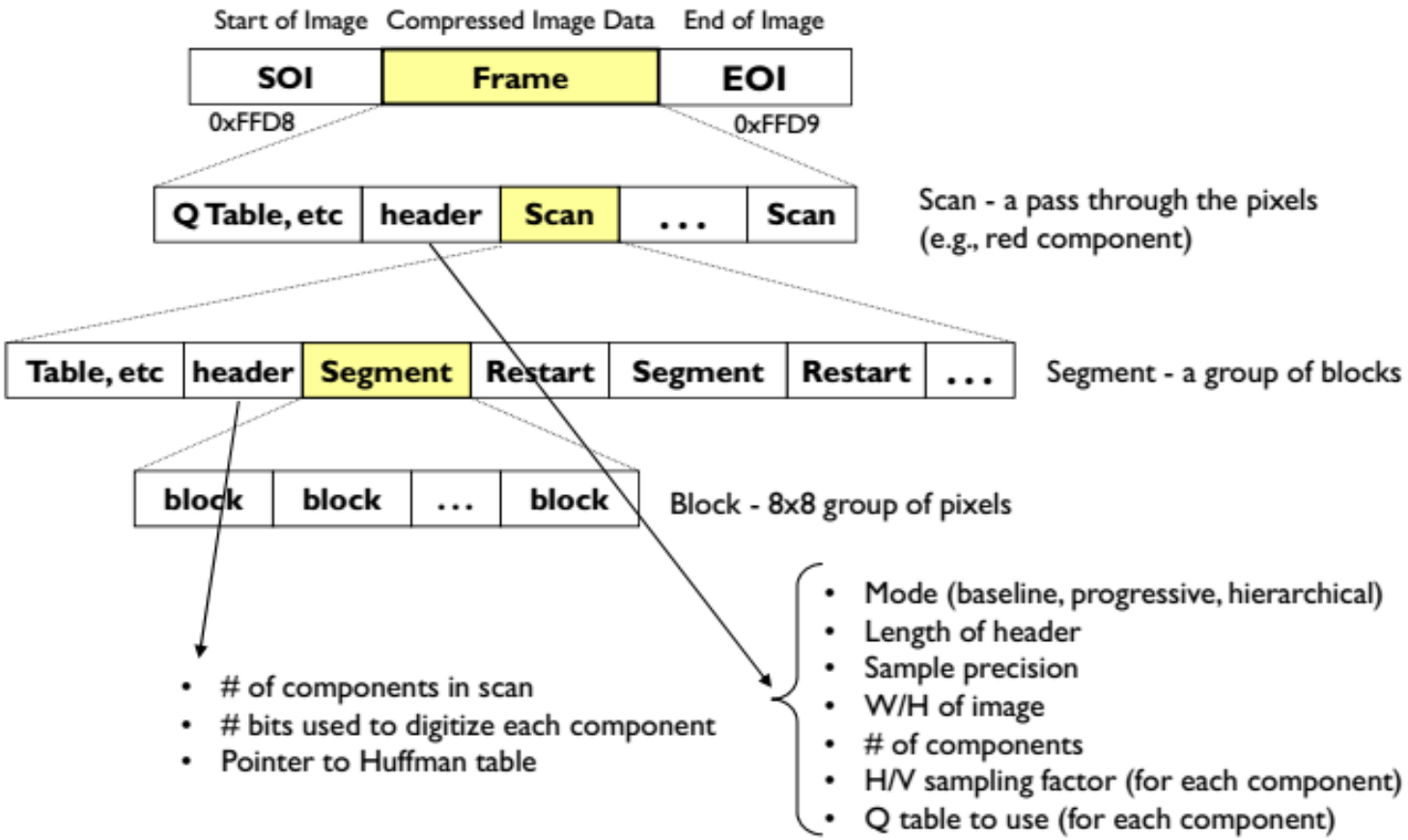
Assume this is the first DC value from a picture!

Encoded sequence

(1110)(001110) (111111110100)(0011) (11111110110)(01111)
(111011)(0) (111110110)(1001) (1100)(0) (11111000)(0) (1100)(0)
(111111111000)(1) (1010)

=> 104 bits! (compared to 8×8×8 = 512 bits)

Formato de Cabeçalho JPEG



UNIVERSIDADE FEDERAL DE PELOTAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

TEC2/TEC4: REDES MULTIMÍDIA (RMM)

Unidade 3
Codificação de Imagem

Prof. Guilherme Corrêa
gcorrea@inf.ufpel.edu.br

DCT de 1 dimensão Inversa

$$pixel(x) = \sqrt{\frac{2}{N}} C(i) \sum_{i=0}^{N-1} DCT(i) \cos \frac{(2x+1)i\pi}{2N}$$

$$= \underbrace{\frac{S(0)}{\sqrt{N}}}_{\text{DC Component}} + \underbrace{\sqrt{\frac{2}{N}} C(i) \sum_{i=0}^{N-1} DCT(i) \cos \frac{(2x+1)i\pi}{2N}}_{\text{AC Component}}$$

$$C(i) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } i = 0 \\ 1 & \text{if } i > 0 \end{cases}$$