Compressão de Imagem

Rivera

Compressão de Imagem

- Formas de diminuir a área de armazenamento dos dados, reduzindo a quantidade de bits para representar os dados (imagem, texto, ou arquivo qualquer).

- Em compressão de imagem define-se como a forma (algoritmos e métodos) de armazenar informações visuais mais compactamente.

Redundâncias na Imagem

Tipos de redundância em imagens:

- **De codificação de tons ou cor**
  - ✓ níveis de cinza ou cores da imagem codificados com mais símbolos de codificação do que o necessário.
- **Inter-pixel**
  - ✓ resultantes das relações geométricas entre os objetos na imagem.
- **Espectral**
  - ✓ valores espectrais, para a mesma posição na matriz de pixels de cada banda, são correlacionados.
- **Psicovisuais**
  - ✓ relacionadas ao fato do sistema visual humano não responder com a mesma sensibilidade a todas as informações visuais.

Compressão de imagens e modelos de cores

- YIQ (para transmissão de televisão);
- RGB (para monitores de computador colorido); CMY (para impressoras coloridas;
- HSI (*Hue*, *Saturation*, *Intensity* ou matiz, saturação, intensidade);
- HSV (*Hue*, *Saturation*, *Value*) ou matiz, Saturação e Valor;
- YCBCR - compressão de imagens (usado no formato de imagens JPEG).

Medição do Desempenho

Medida de desempenho -> taxa de compressão

Tamanho(ImagComp) / tamanho(ImagOrig)

- Técnicas sem perda
  - ✓ quanto maior a taxa de compressão melhor é a técnica de compressão.
- Técnicas com perda
  - ✓ deve-se considerar também a qualidade do sinal ou dado reconstruído.

- Critérios de fidelidade
  - ✓ se a remoção de dados causou perda de informação visual.
  - ✓ Podem ser: quantitativos ou subjetivos.

Critérios de fidelidade objetivos

Sendo *F(x, y)* a imagem original e *G*(*x*, *y*) a imagem reconstruída, tem-se:

Erro Total ou absoluto:

$$e_t = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} \left| G(x,y) - F(x,y) \right|$$

Raiz Quadrada do Quadrado da Média dos Erros:

$$e_{rms} = \sqrt{\left[\frac{1}{MN} \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} \left[G(x,y) - F(x,y) \right]^2 \right]}$$

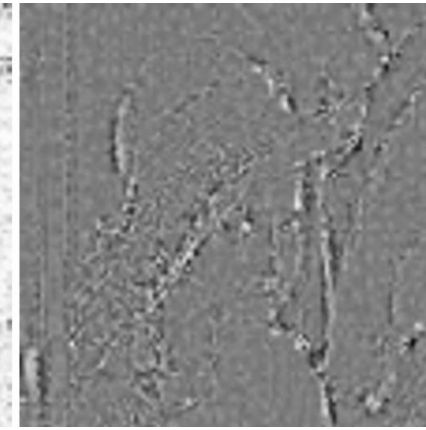
Razão ou Relação Sinal Ruído:

$$SNR_{rms} = \sqrt{\frac{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} G(x,y)^2}{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} e(x,y)^2}} = \sqrt{\frac{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} G(x,y)^2}{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} \left[G(x,y) - F(x,y) \right]^2}}$$

Original

Comprida (fractal) e reconstruída

Diferença absoluta



Diferença relativa

Erro rms= 9,7622

SNR rms 10,4823

PSNR (dB)28,3398

Métodos de Compressão de Imagem

1. Compressão sem perda ou codificação de redundância
    - Preserva todas as informações para reconstrução exata da imagem
    - Explora a redundância entre pixels na codificação
    - Ex. RLE (*Run Lenght Encoding*), LZ (*Lempel Ziv*), LZW (*Lempel Ziv Wech*), algoritmo de Huffman (usadas nos formatos: PCX, PNG, GIF, TIFF).
2. Compressão com perda
    - Há perda de dados durante a compressão da imagem
    - É mais eficiente em relação à área final de armazenamento
    - Não é admissível em aplicações médicas
    - Degradação visual na imagem

Compressão Simétrica e Assimétrica

Classificação quanto ao tempo de compressão e descompressão.

- <u>**Compressão simétrica**</u>: tempo de compressão é igual ao de descompressão
  - ✓ Transformadas de *Wavelets* (WT) e Transformadas de Cossenos (DCT - *Discrete Cosine Transform*).
- <u>**Compressão assimétrica**</u>: tempo de compressão é bem maior que o tempo de descompressão
  - ✓ fractal.

Entropia da Imagem

Seja $A = \{a_1, a_2, ..., a_J\}$ tonalidade de cinza ou tabela de cores de RGB

Entropia permite saber se uma imagem tem redundância

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

4	4	4	4	64	64	128	128
4	4	4	4	64	64	128	128
4	4	16	16	128	128	128	128
4	4	16	16	128	128	128	128

Imagem 4x8=32 *pixels* em *grayscale*

Cor	Total	Probabilidade
4	12	12/32=3/8
16	4	4/32=1/8
64	4	4/32=1/8
128	12	12/32=3/8

Probabilidades para cada nível de cinza

$H(A) = - P(4) * \log_2(P(4)) - P(16) * \log_2 (P(16)) - P(64) * \log_2 (P(64)) - P(128) * \log_2 (P(128))$

$H(A) = -[3/8 * \log_2(3/8) + 1/8 * \log_2(1/8) + 1/8 * \log_2(1/8) + 3/8 * \log_2(3/8)] = 0.81$ *bits/pixel*

Usados 4x8 pixels = 32 pixels ➔ (0.81 bits/pixel) x (32 pixel) = 25 bits

De informação redundante.

Codificação de Huffman

- Redundância de codificação é eliminada
- Codificação de tamanho variável
- Atribui os códigos de tamanhos menores aos níveis de cinza mais prováveis de ocorrer.

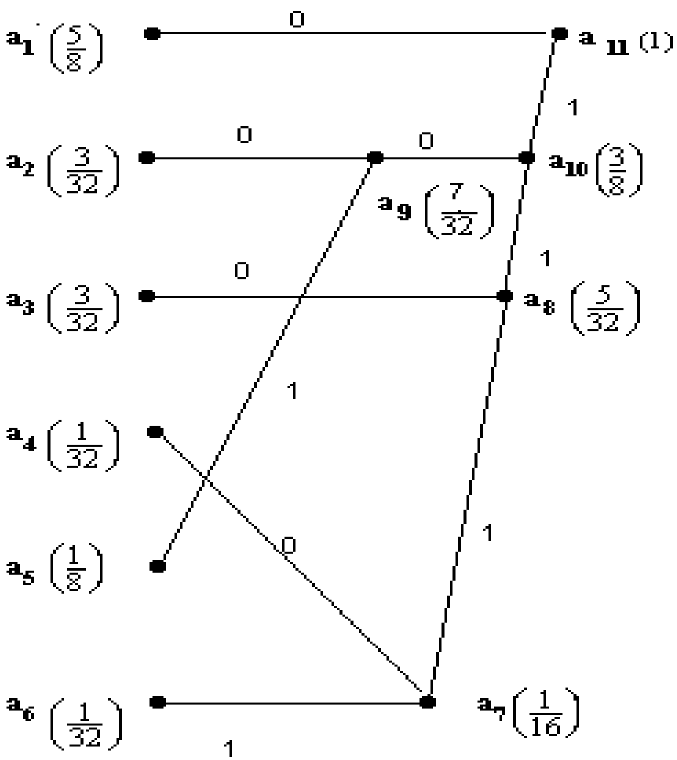
Duas etapas:

1. Redução
    - Cria símbolos juntando dois de menores probabilidades a cada iteração.
2. Codificação
    - Símbolos reduzidos - começando com o de maior probabilidade que será associado ao menor código e voltando para os originais.

Exemplo:

Seja imagem M: 10x10 de 6 tons de cinza (a_1 a_2 a_3 a_4 a_5 a_6)

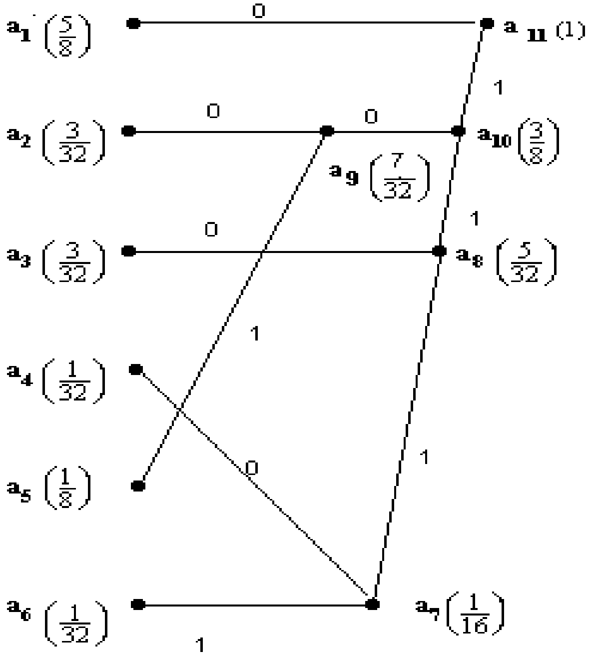
Primeira etapa: redução



1. *Selec. de* (a_1 a_2 a_3 a_4 a_5 a_6) *2 de menor prob.*
   - $p(a_4) + p(a_6) = 1/16 = p(a_7)$
2. *Selec. de* (a_1 a_2 a_3 a_5 a_7) *2 de menor prob.*
   - $p(a_3) + p(a_7) = 5/32 = p(a_8)$
3. *Selec. de* (*a1 a2 a5 a8*) *2 de menor prob.*
   - *p(a2) + p(a5) = 7/32 = p(a9)*
4. *Selec. de* (*a1 a8 a9*) *2 de menor prob.*
   - *p(a8) + p(a9) = 3/8 = p(a10)*
5. *Selec. de* (*a1 a10*) *2 de menor prob.*
   - *p(a1) + p(a10) = 1 = p(a11)*

Segunda etapa da codificação de Huffman

Informação	Probabilidade	Código
$a1$	5/8=20/32	0
$a10$	3/8=12/32	1
$a9$	7/32	10
$a8$	5/32	11
$a5$	1/8=4/32	101
$a2$	3/32	100
$a3$	3/32	110
$a7$	2/32	111
$a4$	1/32	1110
$a6$	1/32	1111



Uma cadeia de códigos: 110 0 100 0 1111 110 0 101 0 1110

É: $a3$ $a1$ $a2$ $a1$ $a6$ $a3$ $a1$ $a5$ $a1$ $a4$

Codificação por LZW

Usa tabela de palavras contendo os símbolos que serão codificados.

Routine LZW_COMPRESS

```
STRING = get input char
WHILE there are still input chars DO
    CHAR = get input char
    IF STRING+CHAR is in the string table THEN
          STRING = STRING+CHAR
    ELSE
         output the code for STRING
         add STRING+CHAR  to the string table
         STRING = CHAR
    END of IF
END of WHILE
output the code for STRING
```

/WED/WE/WEE/WEB/WET

/WED-E-*-B*T

Exemplo:

Input String = /WED/WE/WEE/WEB/WET			
CharInp	CodeOut	NewCodeVal	NewString
/W	/	256	/W
E	W	257	WE
D	E	258	ED
/	D	259	D/
WE	256	260	/WE
/	E	261	E/
WEE	260	262	/WEE
/W	261	263	E/W
EB	257	264	WEB
/	B	265	B/
WET	260	266	/WET
EOF	T		

Decodificação LZW

Routine LZW_DECOMPRESS

Read OLD_COD

output OLD_COD

WHILE there are still input characters DO

Read NEW_COD

STRING = get translation of NEW_COD

output STRING

CHAR = first character in STRING

add OLD_COD + CHAR to the translation table

OLD_COD = NEW_COD

END of WHILE

Input Codes: / W E D 256 E 260 261 257 B 260 T

Input/ newCod	oldCod	STRING/ Ouput	CHAR	New table entry
/	/	/	/	/
W	/	W	W	256 = /W
E	W	E	E	257 = WE
D	E	D	D	258 = ED
256	D	/W	/	259 = D/
E	256	E	E	260 = /WE
260	E	/WE	/	261 = E/
261	260	E/	E	262 = /WEE
257	261	WE	W	263 = E/W
B	257	B	B	264 = WEB
260	B	/WE	/	265 = B/
T	260	T	T	266 = /WET

/WED/WE/WEE/WEB/WET

26/10

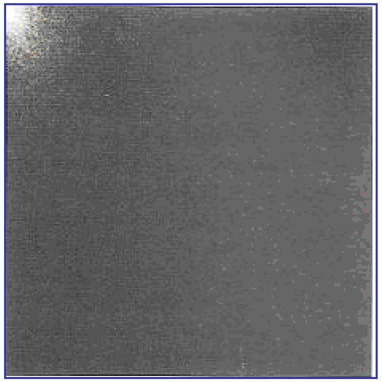
- MARIA
- Javier
- Yuri

Transformada Discreta do Co-seno (DCT)

Transforma discreta de co-senos em 2-D : Espacial → freqüência

$$T[i,j] = c(i)c(j)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} I[y,x]\cos\frac{(2y+1)i\pi}{2N}\cos\frac{(2x+1)j\pi}{2N}$$

onde $$c(i),c(j) = \begin{cases} \sqrt{\frac{2}{N}} & i,j \neq 0 \\ \sqrt{\frac{1}{N}} & i,j = 0 \end{cases}$$



Transformada Inversa IDCT 2-D:

$$I[y,x] = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} c(i)c(j)T[i,j]\cos\frac{(2y+1)i\pi}{2N}\cos\frac{(2x+1)j\pi}{2N}$$

Essa compressão é usada no formato JPEG padrão com valor de N igual a 8.

Compressão por Wavelets

Qualquer função, com período 2π, pode ser reescrita como a soma dos termos da Série de Fourier:

$$a_0 + \sum_{n=1}^{\infty} (a_n \cos nt + b_n \,\mathrm{sen}\, nt)$$

Os coeficientes são calculados por:

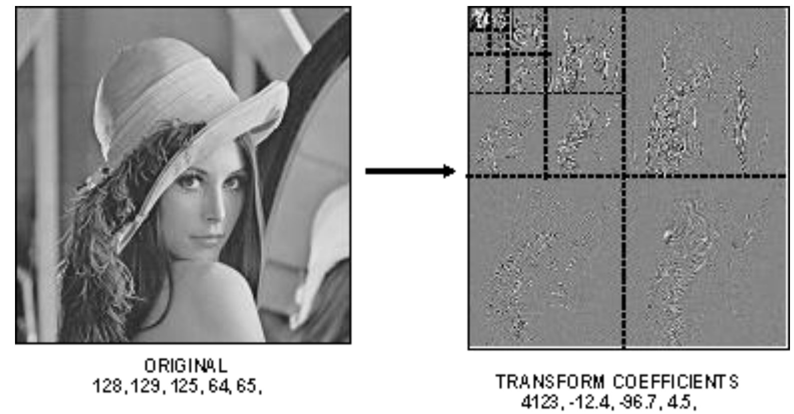
$$a_0 = \frac{2}{T} \int_0^{2\pi} f(t)\, dt$$

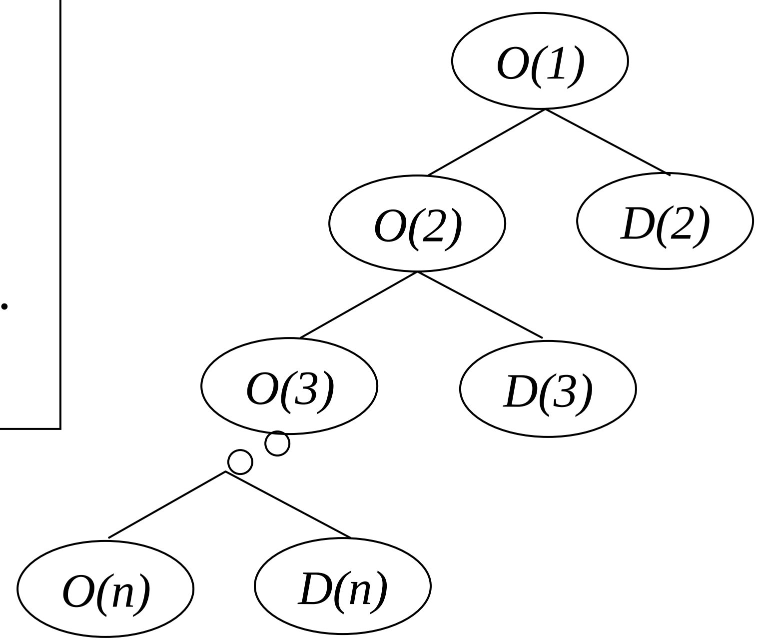
$$a_n = \frac{2}{T} \int_0^{2\pi} f(t) \cos(nt) dt$$

$$b_n = \frac{2}{T} \int_0^{2\pi} f(t)\, \mathrm{sen}(nt) dt$$

Análise de *Wavelet*

- Ferramenta matemática para decomposição em nível hierárquico em aproximações (O) e detalhes (D).
- O nível hierárquico em escala diática (formado por potência de 2).
- Descrição de uma função em termos globais, mais termos que variam de detalhes globais até detalhes finos.
- A função em questão pode ser uma imagem, uma curva ou uma superfície.

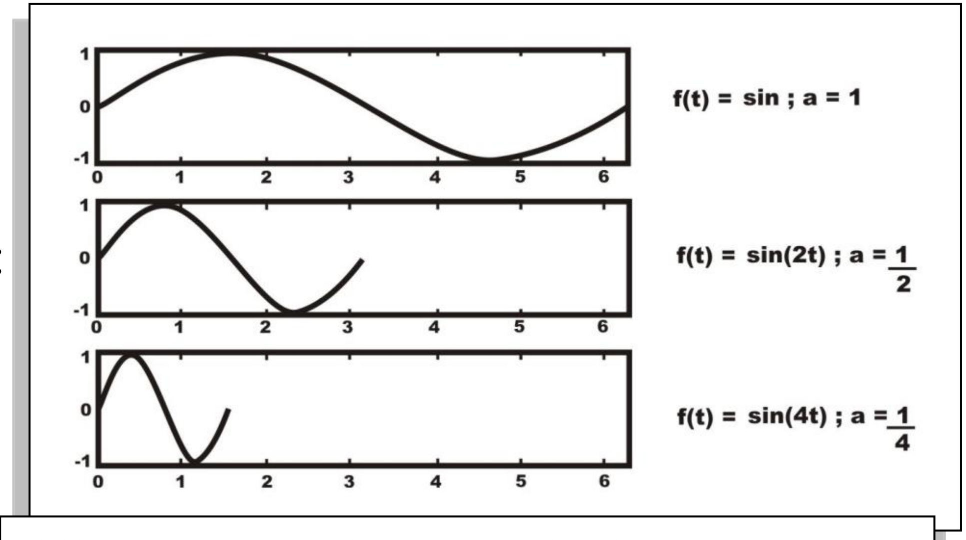




Transformada de *Wavelet* Contínua (TWC)

A Transformada de *Wavelets* contínua em *F(a,b)* é:

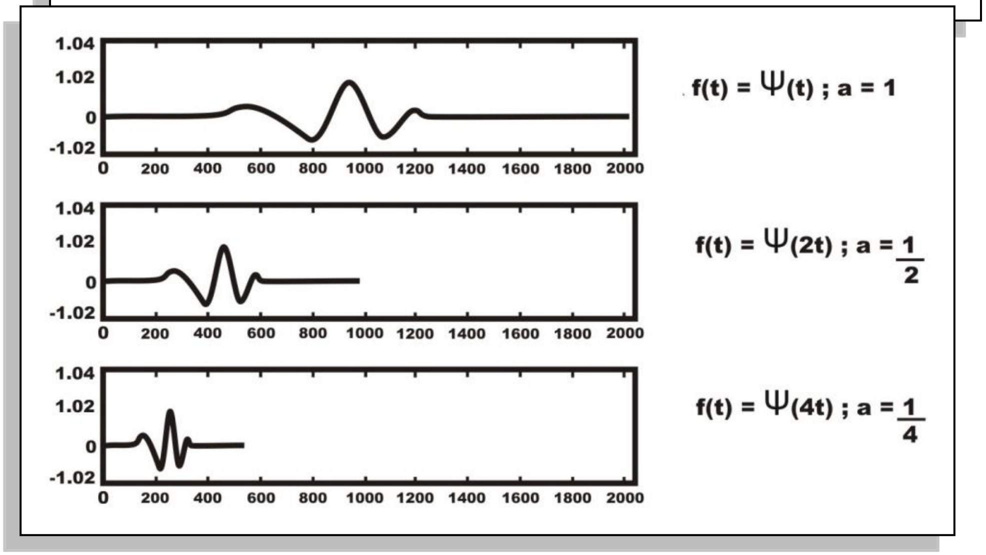
$$F(a,b) = \int f(t)\, \Psi_{a,b}\ (t)\, dt$$



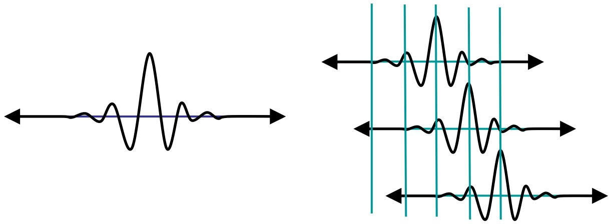
A função $\Psi_{a,b}(t)$ é denominada *wavelet*:

$$\Psi_{a,b}(t) = \frac{1}{\sqrt{a}}\,\Psi\!\left(\frac{t-b}{a}\right),\ a \neq 0\,,\ b \in \Re$$

As funções *wavelets* são derivadas segundo os critérios:



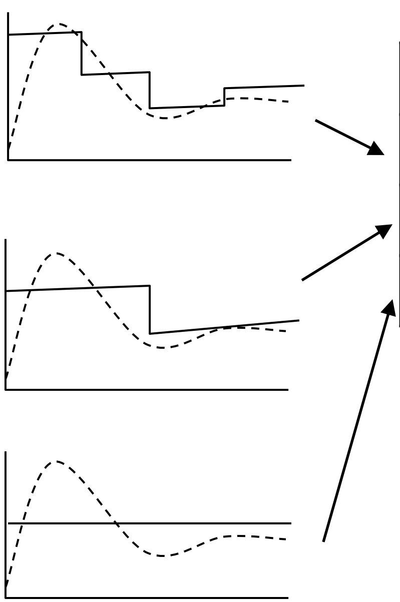
$$C_{\Psi} = 2\,\pi \int |u|^{-1} \left|\hat{\Psi}(u)\right|^{2} du < \infty$$



Transformada de *Wavelet* Discreta

$$\Psi_{a\,,\,b}(t)=\frac{1}{\sqrt{a}}\,\Psi\!\left(\frac{t-b}{a}\right),\quad a=2^{\,j},\, b=k\;2^{\,j},\quad (j,k)\in Z^{\,2}$$

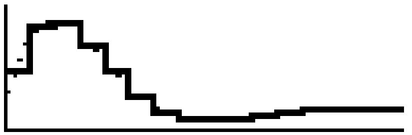
Transformada Wavelet Haar Unidimensional



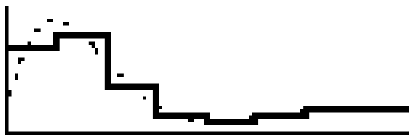
Resolução	Valor Aprox.	Coef. Detalhes
4	[9 7 3 5]	
2	[8 4]	[1 -1]
1	[6]	[2]

[6 2 1 -1]

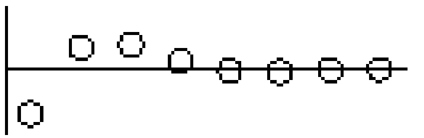
Seqüência de aproximação e coeficientes de detalhes.



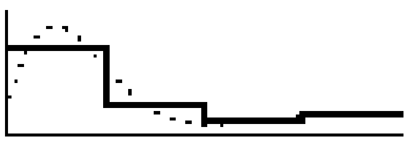
Aproximação V^4



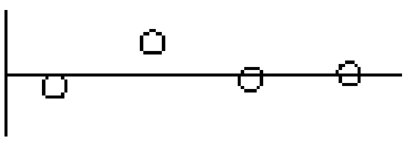
Aproximação V^3



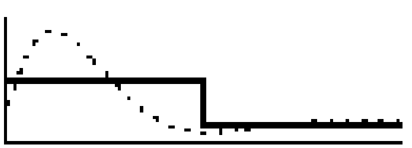
Coeficientes de detalhes W^3



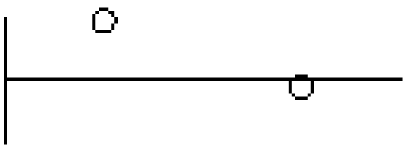
Aproximação V^2



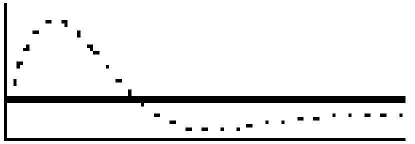
Coeficientes de detalhes W^2



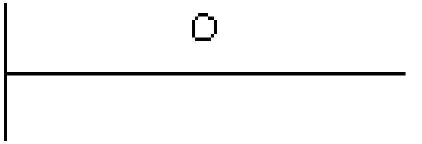
Aproximação V^1



Coeficientes de detalhes W^1



Aproximação V^0



Coeficientes de detalhes W^0

Espaço de Imagem

$$V^0 \subset V^1 \subset V^2 \ldots$$

Base de V^j

$$\phi_i^j(x) = \phi(2^j x - i)$$

Espaço de Detalhes

$$W^0 \oplus V^0 = V^1$$

$$W^1 \oplus V^1 = V^2$$

Base de W^j

$$\psi_i^j(x) = \psi(2^j x - i)$$

Bases Haar

Base mãe $\phi(x) = \begin{cases} 1 & 0 \le x \le 1 \\ 0 & outro \end{cases}$

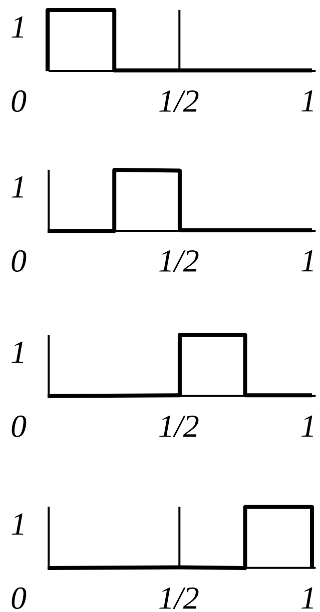
$$\phi_i^j(x) = \phi(2^j x - i)$$

I(x) = 9 x

+ 7 x

+ 3 x

+ 5 x



wavelet mãe $\psi_i^j(x) = \psi(2^j x - i)$

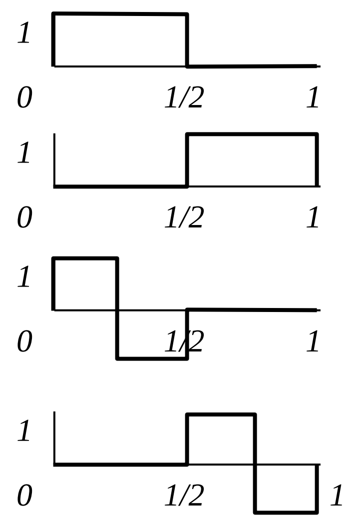
$$\psi(x) = \begin{cases} 1 & 0 \le x \le 1/2 \\ -1 & 1/2 < x \le 1 \\ 0 & Outro \end{cases}$$

I(x) = 8 x

+ 4 x

+ 1 x

- 1 x



$$\boxed{I(x) = c_0^2\phi_0^2(x) + c_1^2\phi_1^2(x) + c_2^2\phi_2^2(x) + c_3^2\phi_3^2(x)}$$

$$\boxed{I(x) = c_0^1\phi_0^1(x) + c_1^1\phi_1^1(x) + d_0^1\psi_0^1(x) + d_1^1\psi_1^1(x)}$$

Normalidade

Uma função base u(x) é normalizada se $\langle u | u \rangle = 1$

Haar normalizado:

$$\phi_i^j(x) := 2^{j/2}\,\phi\left(2^j x - i\right), \quad i = 0, \ldots, 2^j - 1$$

$$\psi_i^j(x) := 2^{j/2}\,\psi\left(2^j x - i\right), \quad i = 0, \ldots, 2^j - 1$$

Ex. para [6 2 1 -1], se tornam normalizados:

$$\begin{bmatrix} 6 & 2 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix}$$

Algoritmo de Transformada Wavelet Haar

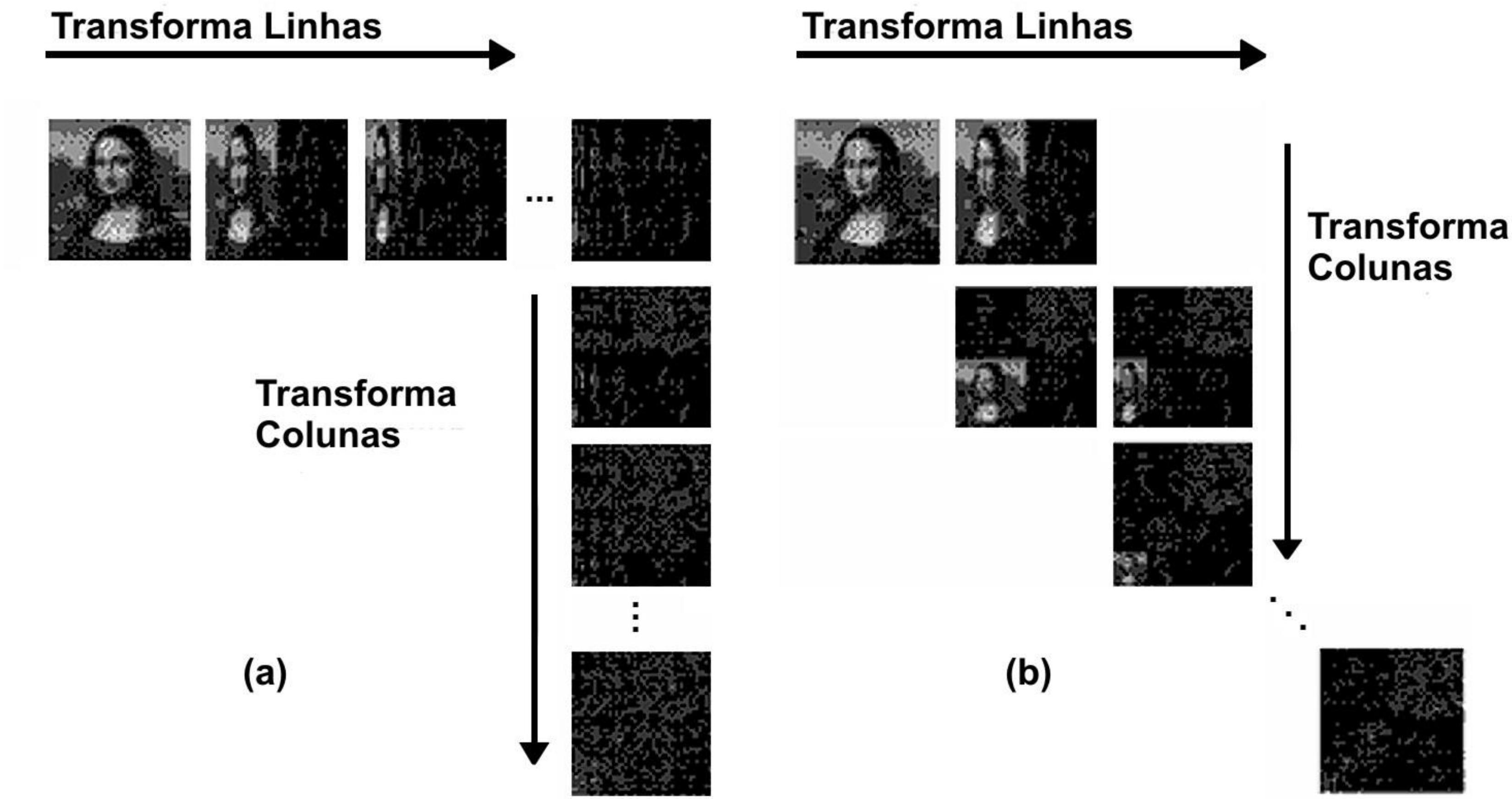
```
Decomposi ( C[1...2^j ])
   C ← c / sqrt (2 ^ j )     // normaliza coef inp
   g ← 2 ^ j
   WHILE g >= 2  DO
      DecomposiStep ( C [1 .. g ])
      g ← g / 2
   END
```

```
DecomposiStep ( C[1...2^j ])
   FOR i = 1 .. 2 ^ (j-1)
      C' ← ( C[2i – 1] + C[2i] ) / sqrt (2)
      C'[2^(j-1) + 1] ← (C[2i – 1] - C[2i] ) / sqrt (2)
   END
   C ← C'
```

```
Reconstruc ( C[1...2^j ])
   g ← 2
   WHILE g =< 2 ^ j   DO
      ReconstrucStep ( C [1 .. g ])
      g ← 2g
   END
   C ← C sqrt ( 2 ^ j )   // sem normalização
```

```
ReconstrucStep ( C[1...2^j ])
   FOR i = 1 .. 2 ^ (j-1)
      C'[ 2i – 1 ] ← ( C[ i ] + C[2 ^ (j-1) + i] ) / sqrt (2)
      C'[2i] ← (C[ i ] - C[2 ^ (j –i) + i ] ) / sqrt (2)
   END
   C ← C'
```

Transformada de *Wavelet* de Haar bidimensional



(a) Decomposição padrão (b) Decomposição não padrão.

Compressão

O objetivo da compressão é expressar um conjunto inicial de dados usando outro conjunto menor, com ou sem perda de informação.

Seja a imagem f (x) expressa pela soma de funções base :

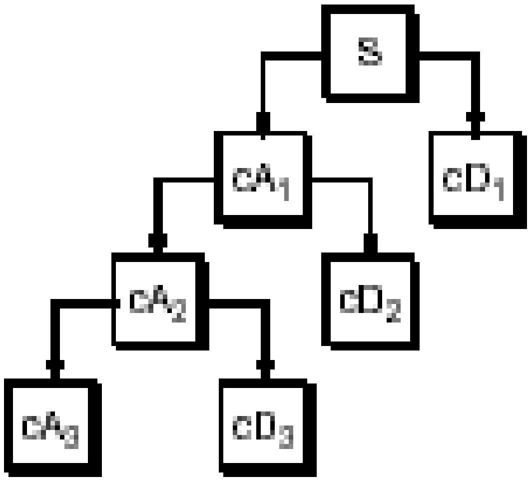
$$f(x) = \sum_{i=1}^{m} c_i u_i(x)$$

Com *m* coeficientes c_i.

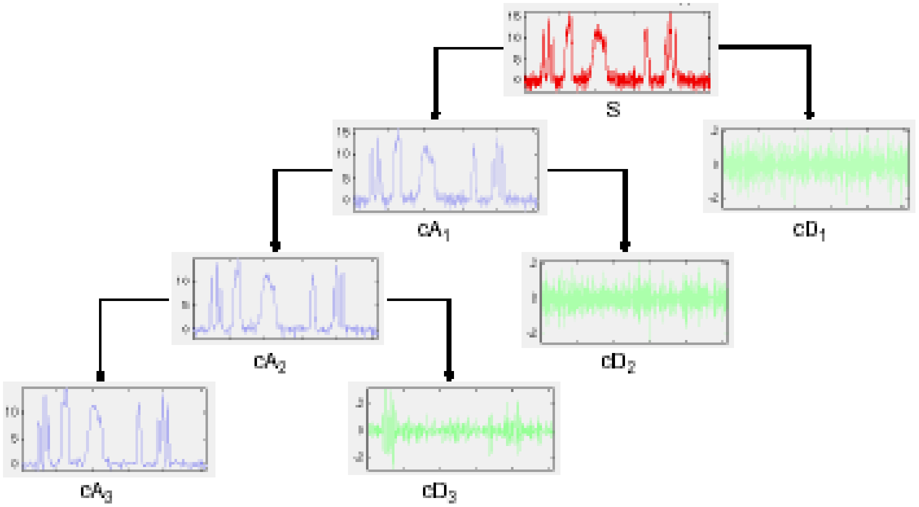
A função que aproxima *f(x)*, com menos coeficientes:

$$\tilde{f}(x) = \sum_{i=1}^{\tilde{m}} \tilde{c}_i \, \tilde{u}_i(x) \cong f(x)$$

Aproximações e Detalhes



Árvore de Decomposição *Wavelet*



Árvore de Decomposição *Wavelet* de um sinal