Sisteme avansate de codare și compresie a datelor multimedia (SACCDMM)

- Curs 4 -

Algoritmi de compresie cu/fără pierderi

Intelligent and multimodal image processing and analysis group (IMIPA), Communications Departament, ETTI, TUCN,
E-mail: Camelia.Florea@com.utcluj.ro; Phone: +4 0264 401285; Address: C. Daicoviciu, 15, room 433, Cluj-Napoca, RO.

Compresie CU sau FĂRĂ pierderi

Compression can be categorised in two broad ways:

Lossless Compression: after decompression gives an exact copy of the original data.

Example: Entropy encoding schemes (Shannon-Fano, Huffman coding), arithmetic coding, LZW algorithm (used in GIF image file format).

Lossy Compression: after decompression gives ideally a "close" approximation of the original data, ideally perceptually lossless.

Example: Transform coding — FFT/DCT based quantisation used in JPEG/MPEG differential encoding, vector quantisation.

De ce Compresie cu PIERDERI

- Lossy methods are typically applied to high resoultion audio, image compression.
- Have to be employed in video compression (apart from special cases).

Basic reason:

- Compression ratio of lossless methods (*e.g*. Huffman coding, arithmetic coding, LZW) is not high enough for audio/video.
- By cleverly making a small sacrifice in terms of fidelity of data, we can often achieve very high compression ratios.
  - Cleverly = sacrifice information that is psycho-physically unimportant.

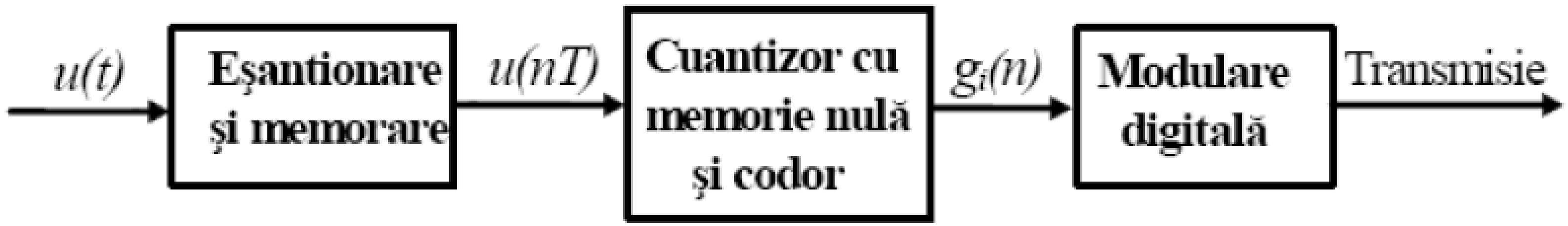
Algoritmi de compresie fără pierderi

- Semantic-based coding/source coding
  - Repetitive Sequence Suppression.
  - Run-Length Encoding (RLE).
  - Pattern Substitution.
  - Entropy Encoding:
    - Shannon-Fano Algorithm
    - Human Coding
    - Arithmetic Coding
  - Lempel-Ziv-Welch (LZW) Algorithm.
- Some broad methods that exist:
  - Diferential Encoding
  - Vector Quantisation
  - Transform Coding

Algoritmi de compresie fără pierderi

- setul original de date poate fi refăcut complet
- există aplicaţii unde NU se acceptă pierderi
  - Imagini medicale
  - Imagini bitonale
    - desene, scrisori, ziare, hărți și alte documente

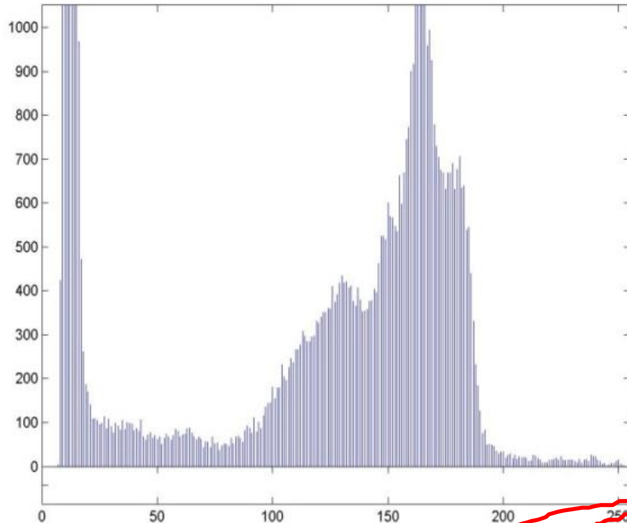
Modularea impulsului în cod (PCM)

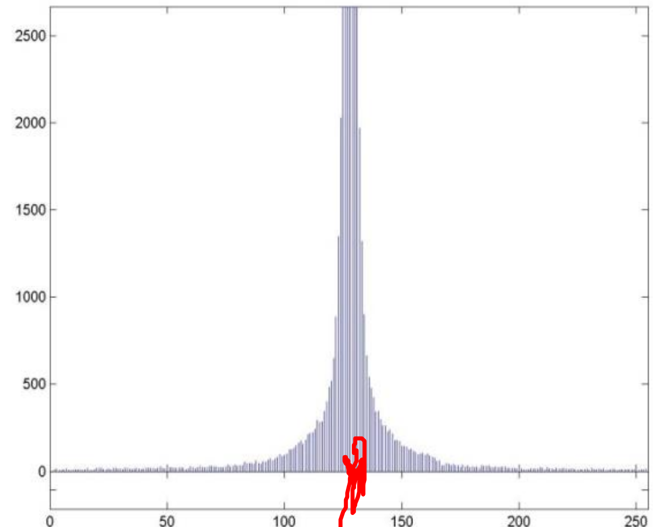


- Semnalul este eșantionat, cuantizat și codat
- Ieșirea cuantizorului este codată cu cuvinte binare de lungime fixă (B biți, tipic 8 biți/cuvant)

Codarea diferenţială

DPCM

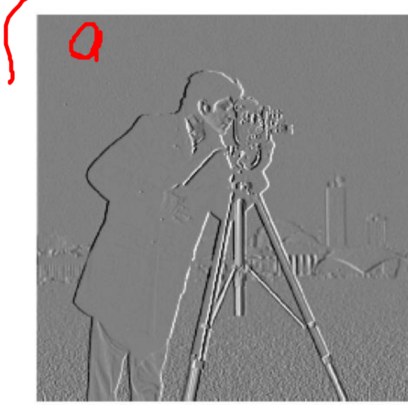




128

- codarea diferenţială
  - îmbunătăţirea ratei de compresie
  - algoritm de preprocesare

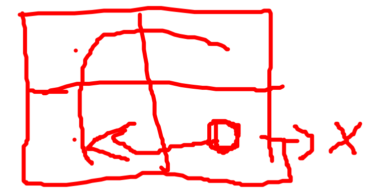
= modificarea ratei de apariţie simbolurilor astfel încât să se obţină distribuţie mai eficientă pentru codare



Codarea predictivă fără pierderi

- Codare

$$e(n) = f(n) - \hat{f}(n)$$



- Decodare

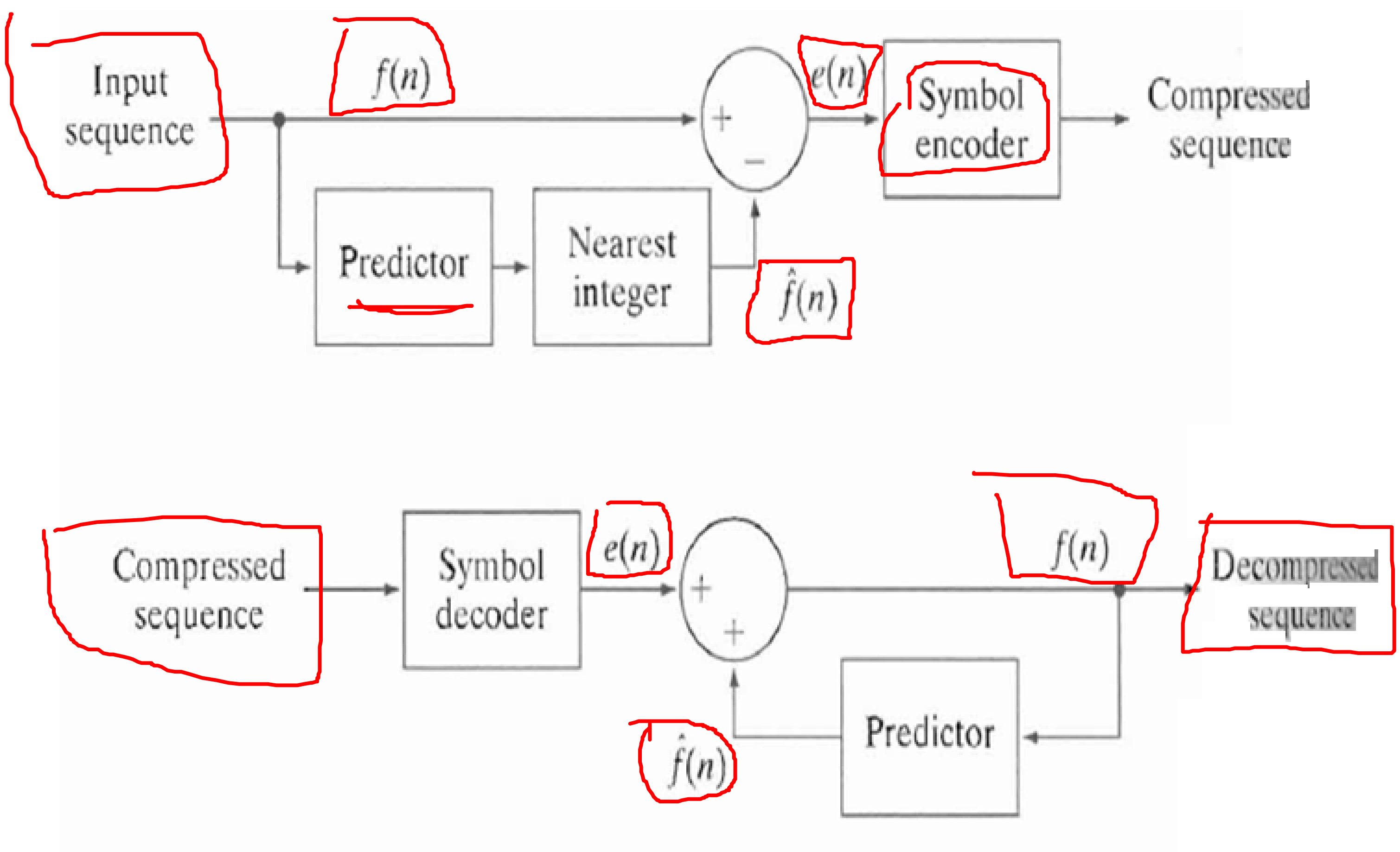
$$f(n) = e(n) + \hat{f}(n)$$

- Predictorul – combinaţie liniară a *m* eşantioane anterioare
  - *m* – ordinul predictorului
  - *ROUND* – cel mai apropiat întreg
  - α_i – coeficienţii de predicţie

$$\hat{f}(n) = \text{round}\left[\sum_{i=1}^{m} \alpha_i f(n-i)\right]$$

Codarea predictivă fără pierderi

- Codorul şi decodorul – acelaşi *predictor*



Codarea predictivă fără pierderi

- 1-D – linia curentă
- 2-D – linia curentă şi anterioară
- 3-D – imaginea curentă şi imaginea anterioară

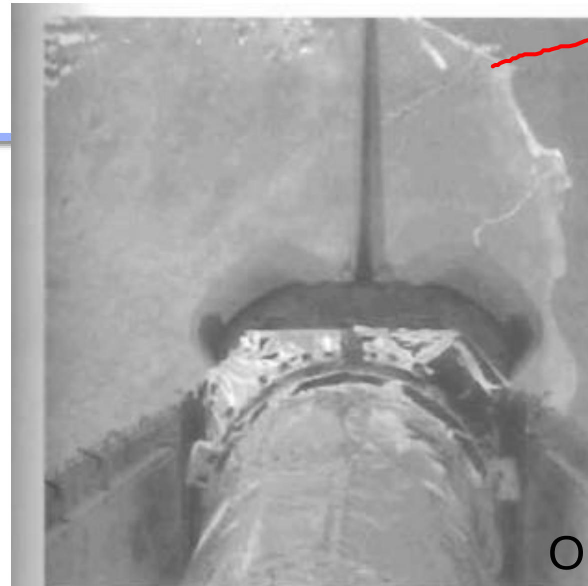
- Pentru imagini (x, y)
  - Dacă
    - m=1, α=1
    - Codarea diferenţială

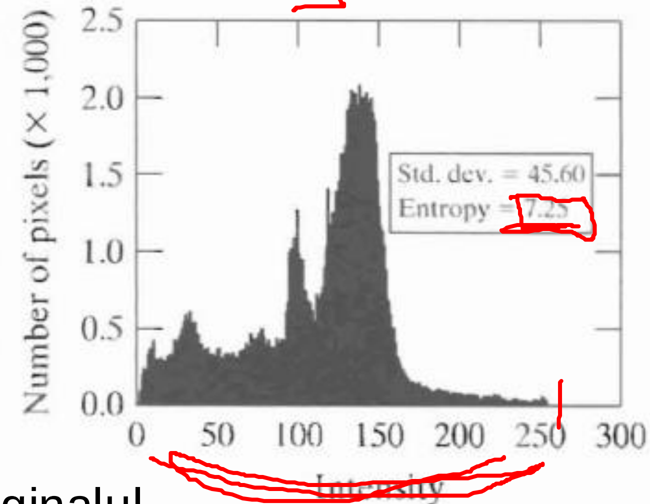
$$\hat{f}(x, y) = \text{round}\left[\sum_{i=1}^{m} \alpha_i f(x, y-i)\right]$$

$$\hat{f}(x, y) = \text{round}\left[\alpha f(x, y-1)\right]$$

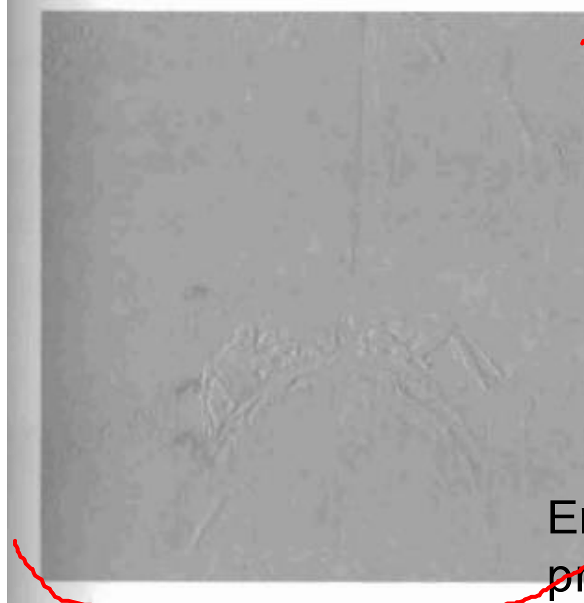
1-D

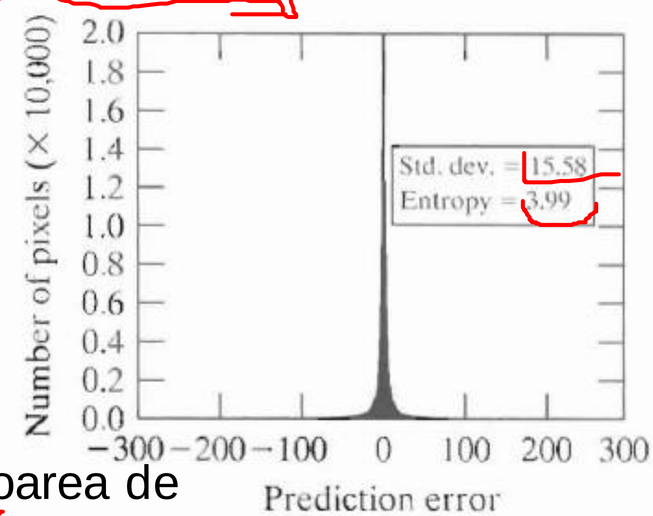
- Imaginea eroarea de predictie
  - scalare pe 128
- Se reduce redundanţa spaţială
- Std. dev.
  - 45.6 la original
  - 15.58 la eroarea de predicţie
- Se poate obţine un C=8/3.99 = aprox. 2





Originalul





Eroarea de predicţie

$m' = 1$, $d_i = 1$

3-D

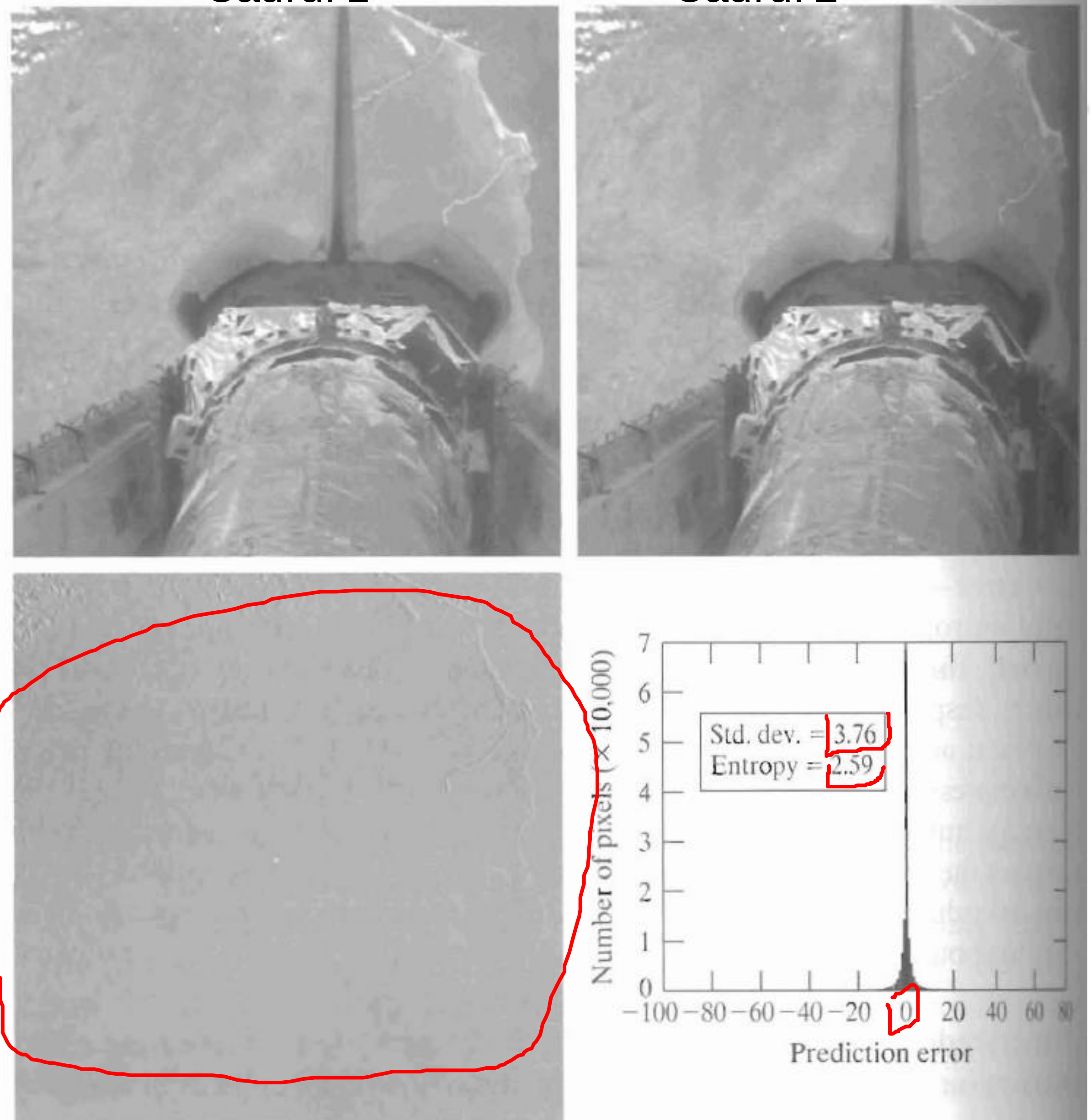
- Scalare pe 128
- Std. dev. – 3.76
- Entropia 2.59
- Se poate obţine un

C=8/2.59 = aprox. 3.1:1

$$\hat{f}(x, y, t) = \text{round}\left[\alpha f(x, y, t-1)\right]$$

$$e(x, y, t) = f(x, y, t) - \hat{f}(x, y, t)$$

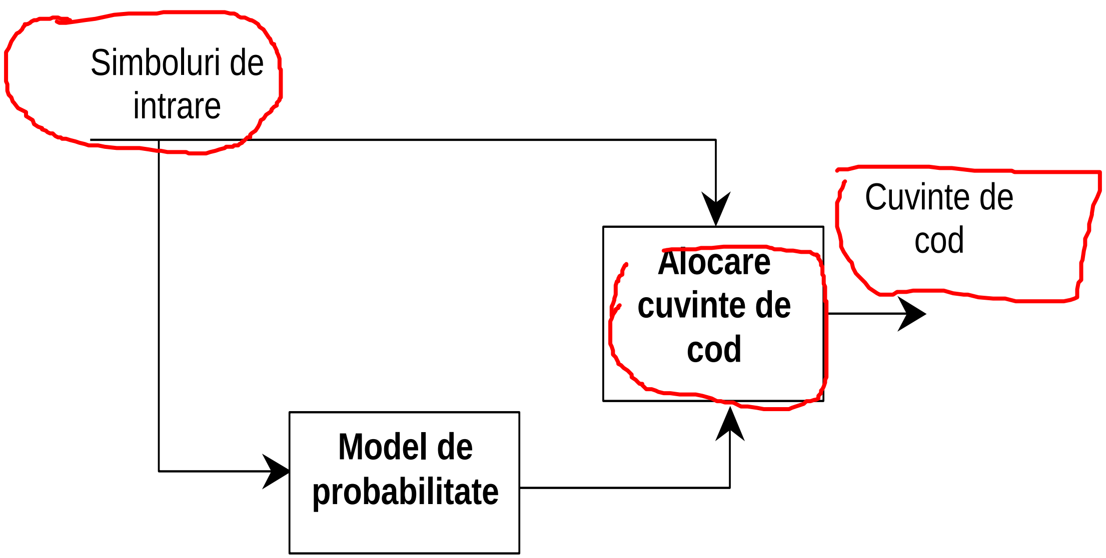
Cadrul 1 Cadrul 2



Eroare de predicţie

Arhitectura codorului entropic

- împărţirea mesajului în simboluri
  - 256x256 pixeli = 1 simbol lungime 64000 pixeli
  - 256x256 pixeli = 256x256 simboluri intre 0 – 255



Codarea Huffman

- **simboluri**
  - probabilitate mare – cod lungime mica
  - probabilitate mica – cod de lungime mai mare
  - tabel coduri Huffman – lungime *L*

Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6 0
a_6	0.3	0.3	0.3	0.3 00	0.4 1
a_1	0.1	0.1	0.2 010	0.3 01	
a_4	0.1	0100 0.1	0.1 011		
a_3	0.06	0.1 0101			
a_5	0.04				

Codarea Huffman

Original source			Source reduction							
Symbol	Probability	Code	1		2		3		4	
a_2	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
a_6	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
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								

- Entropia sursei este 2.14 biţi/pixel

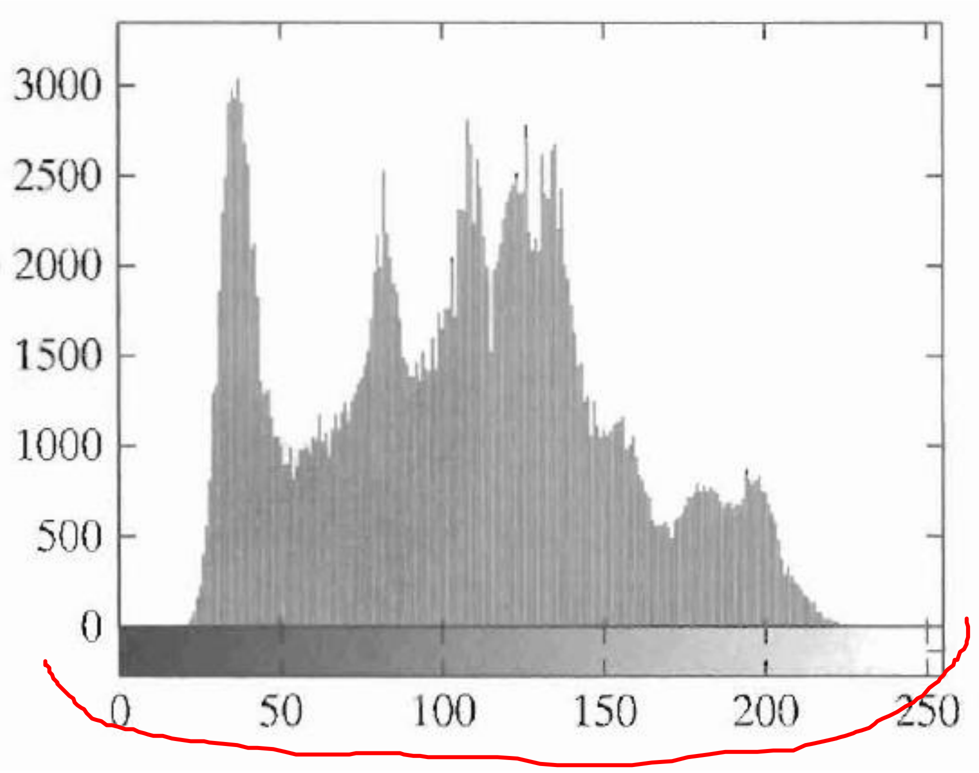
H(S) = -0.4*log2(0.4)-0.3*log2(0.3)-0.1*log2(0.1)-0.1*log2(0.1)-0.06*log2(0.06)-0.04*log2(0.04) = 2.14

- Evident **cod unic**: 010100111100 – a3 a1 a2 a2 a6
- Lungimea medie pe simbol = 2.2 bits/pixel

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

Codarea Huffman

- 515x515x8 biţi/pixel



Codarea Huffman

- Entropia = 7.3838 biţi/pixel
- Lavg = 7.428 biţi/pixel (Huffman - Matlab)

- Diferenţa Lavg-Entropia
  - este de 512x512x(7.428-7.3838),

    => adică 11587 biţi – 0.6%

- C = 8/7.428 = 1.077

- R= 1 - 1/1.077 = 0.0715 (7.15% se reduce prin codare)

Codarea Huffman

- **cod Huffman trunchiat**
  - se alege $L_1 < L$; → 256
  - primele L_1 simboluri se codeaza Huffman
  - celelalte simboluri – cod_prefix+cod lungime fixa

$L_1 = 50$

- **cod Huffman modificat**
  - se alege $L_1 < L$;
  - primele L_1 simboluri se codeaza Huffman
  - celelalte simboluri – cod_prefix (catul q) +cod terminator (codul Huffman al restului j)

$$i = qL_1 + j, \qquad 0 \le q \le Int\left[\frac{(L-1)}{L_1}\right], \qquad 0 \le j \le L_1 - 1$$

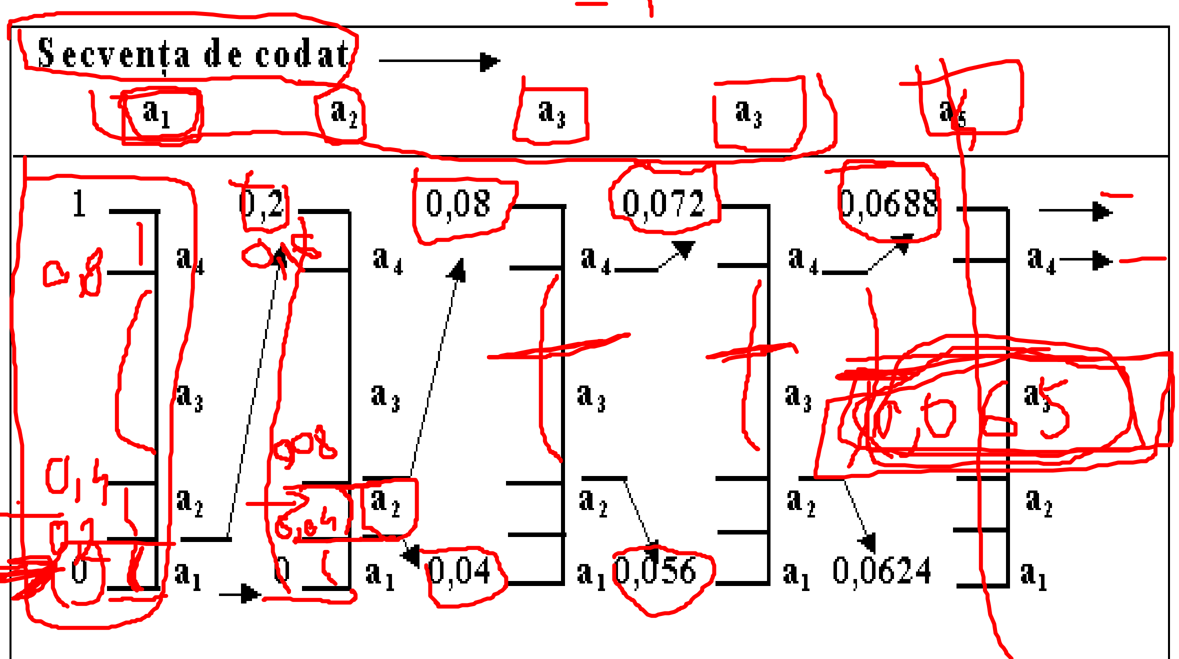
$i = 52 = 1 \cdot 50 + 2$

Codarea artimetică

- nu exista corespondenta intre simbolurile sursei si cuvintele de cod
- cuvant de cod = interval de numere reale din (0,1)
- cu cat numarul de simboluri din mesaj creste -> scade intervalul de reprezentare
- fiecare simbol reduce intervalul conform probabilitatii de aparitie

Codarea aritmetică

Simbol sursă	Probabilitate	Interval inițial
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)



Codarea RLC

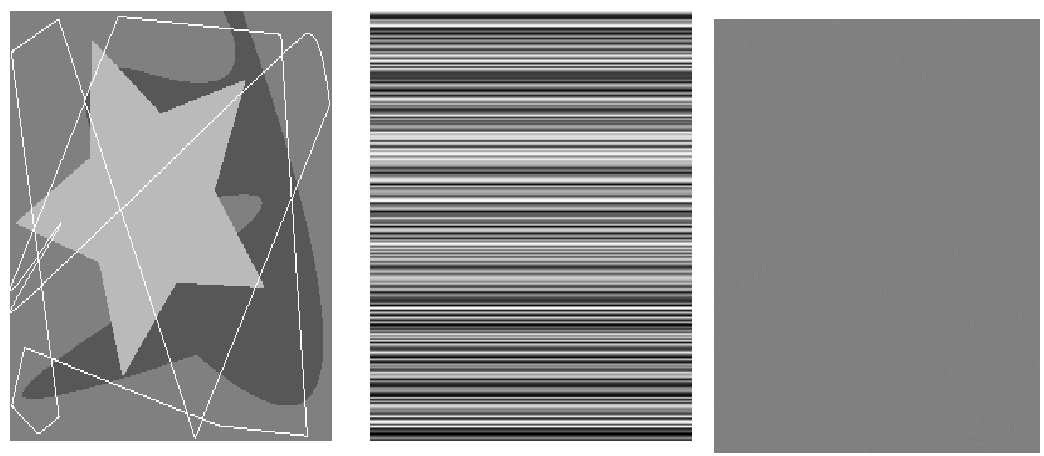
- Imagini în care se repetă intensitatea pe rânduri/coloane
- surse binare de simboluri cu *0* sau *1*
- perechi de lungimi de curse – **start** nouă intensitate, **lungime**
- grafice, documente tiparite, imagini binare *(p (0) – mare aproape de 1)*
- 1000000100000000111110000000
- 11 60 11 80 51 70

Codarea RLC

- BMP – fără codare şi cu codarea RLC

- Figura a
  - BMP-fără codare- 26**3.244** bytes
  - BMP-cu codare- 26**7.706** bytes
  - Mai mare cu codare!!! C=0.98

- Figura c
  - C=1.35



Codarea RLC fingerprint

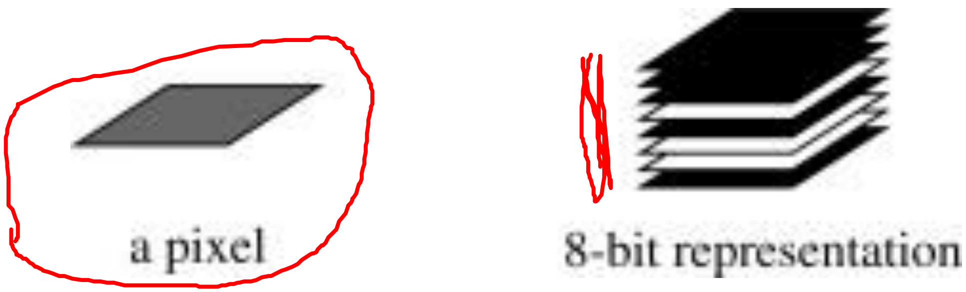
JPEG
DCT

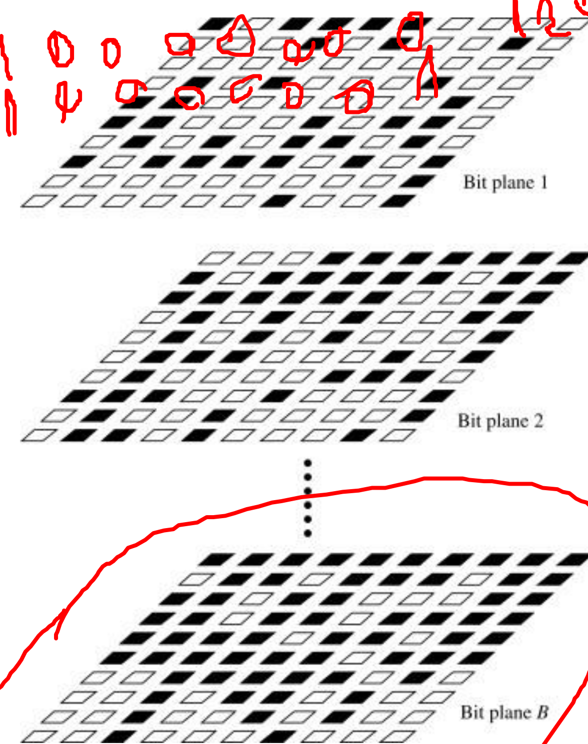
- numar mic de coeficienti
- dupa DCT, Wavelet etc.
- 107-254 – coef. Nenuli
- daca |coef|>73 se transmite cod escape urmat de valoarea coef.
- 1-100 curse de zero
- daca *cursa de zero>100*, se tramsmit 105, 106 si valoarea cursei de zero

Simbol	Valoare
1	cursă de zero de lungime 1
2	cursă de zero de lungime 2
...	...
100	cursă de zero de lungime 100
101	simbol escape pentru index pozitiv 8 biți
102	simbol escape pentru index negativ 8 biți
103	simbol escape pentru index positiv 16 biți
104	simbol escape pentru index negativ 16 biți
105	simbol escape pentru cursă de zero 8 biți
106	simbol escape pentru cursă de zero 16 biți
107	valoare coeficient -73
108	valoare coeficient -72
...	...
180	neutilizat; se utilizează simbolul 1
...	...
253	valoare coeficient 72
254	valoare coeficient 73

Codarea planurilor de biți

Jpeg 2000





- O imagine cu 256 nivele de gri, codată cu 8 biți/pixel, poate fi considerată ca un set de 8 plane de 1 bit (planul MSB(0)… planul LSB (7)) și fiecare dintre acestea este codata RLC

- Metoda poate atinge compresii de pană la 1.5-2, dar este sensibilă la erorile canalului, dacă cele mai semnificative plane de biți nu sunt protejate corespunzător

Algoritmi de codare cu pierderi

- **codarea predictivă**
  - codarea predictivă cu pierderi
  - modulaţia Delta

- **codarea pe blocuri de pixeli**
  - alg. de codare spatială pe blocuri
  - codarea prin transformări

Codarea predictivă

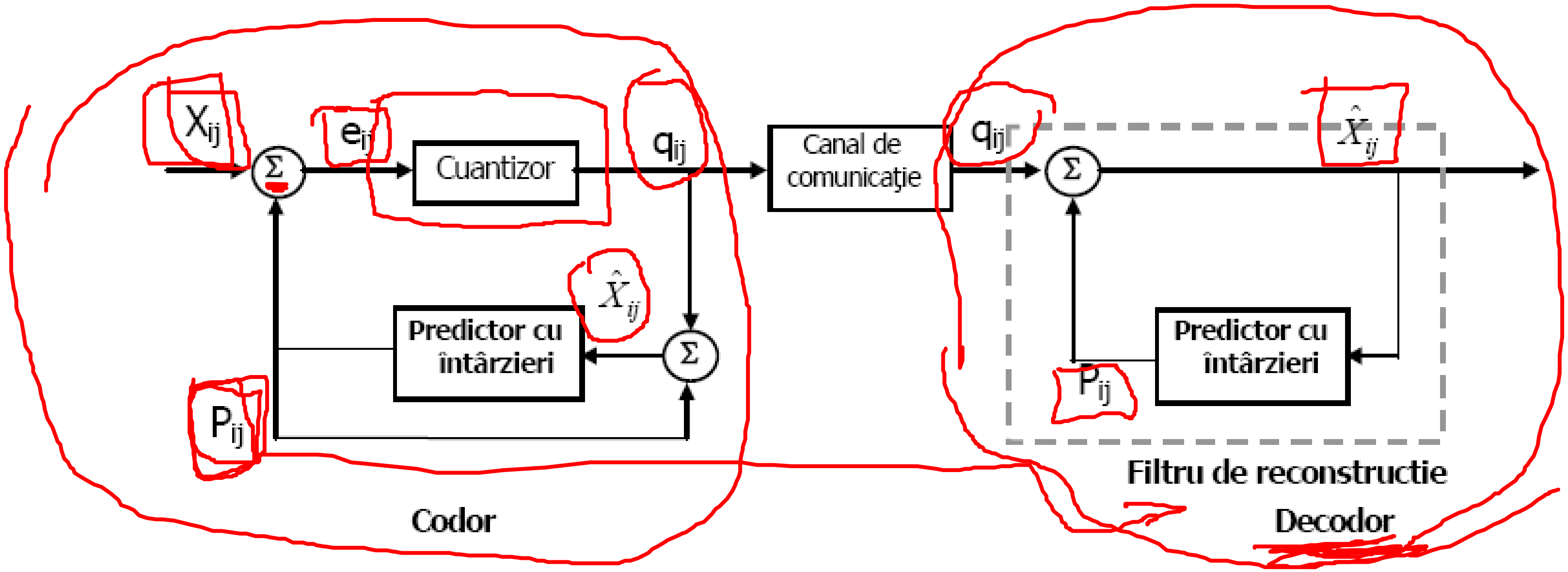
- Eşantioanele care sunt luate în considerare în cazul codării predictive pot fi fie
  - din domeniul spaţial
  - fie din domeniul frecvenţă.

Jpeg → DCT → → coef DC DPCM

- Unul dintre cei mai simplii algoritmi de codare predictivă este:
  - ***codarea DPCM – Differential Pulse Code Modulation***

Codarea predictivă cu pierderi

- Se adaugă un cuantizor
- Eroarea de predicţie va fi mapată într-un domeniu limitat
- Cuantizorul este cel care implică pierderi
- Previne propagarea erorilor la decodor



$\hat{X}_{i-1,j-1}$	$\hat{X}_{i-1,j}$	
$\hat{X}_{i,j-1}$	$\hat{X}_{i,j}$	

Exemplu de predictor pentru codarea DPCM

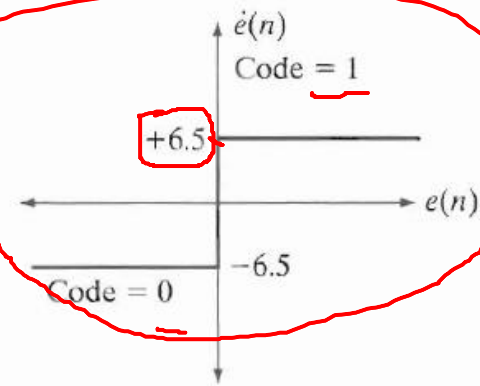
$$P_{i,j} = w_1 X_{i,j-1} + w_2 X_{i-1,j-1} + w_3 X_{i-1,j}$$

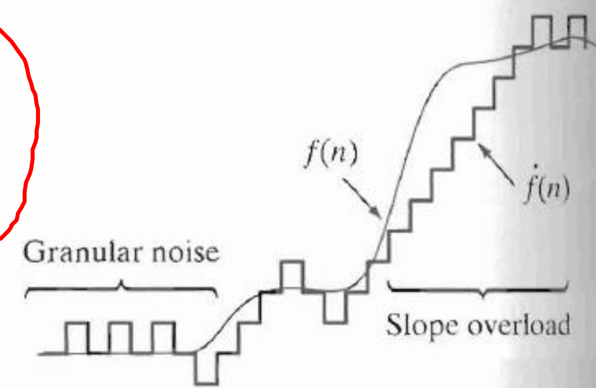
$$\hat{X}_{i,j} = P_{i,j} + q_{i,j}$$

Modulaţia Delta

- Caz particular al codării predictive cu pierderi
- Probleme
  - Depăşirea de pantă
  - Zgomotul de granularitate
- Variante de rezolvare
  - modulaţia adaptivă pe trei stări (±1, 0) datorită faptului că 65-85% sunt pixeli de nivel

$$\hat{f}(n) = \alpha \dot{f}(n-1)$$





Input		Encoder				Decoder		Error
n	$f(n)$	$\hat{f}(n)$	$e(n)$	$\dot{e}(n)$	$\dot{f}(n)$	$\hat{f}(n)$	$\dot{f}(n)$	$f(n) - \dot{f}(n)$
0	14	—	—	—	14.0	—	14.0	0.0
1	15	14.0	1.0	6.5	20.5	14.0	20.5	−5.5
2	14	20.5	−6.5	−6.5	14.0	20.5	14.0	0.0
3	15	14.0	1.0	6.5	20.5	14.0	20.5	−5.5
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
14	29	20.5	8.5	6.5	27.0	20.5	27.0	2.0
15	37	27.0	10.0	6.5	33.5	27.0	33.5	3.5
16	47	33.5	13.5	6.5	40.0	33.5	40.0	7.0
17	62	40.0	22.0	6.5	46.5	40.0	46.5	15.5
18	75	46.5	28.5	6.5	53.0	46.5	53.0	22.0
19	77	53.0	24.0	6.5	59.6	53.0	59.6	17.5
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.

Codarea pe blocuri de pixeli

- **algoritmi de codare spaţială pe blocuri**
  - gruparea pixelilor în blocuri
  - metode clasice de compresie spaţială
  - compresie mai bună decât varianta clasică

- **algoritmi de codare prin transformări**
  - gruparea pixelilor pe blocuri
  - transformarea într-un alt domeniu (frecvenţă)
  - DFT, DCT, DST, Hadamard

Lempel-Ziv-Welch (LZW)

- Este o tehnică de compresie foarte populară
- Folosită în fișiere: GIF (LZW), Adobe PDF (LZW), compresie UNIX (LZ Only)

- Exemplu: codare - Oxford Concise English Dictionary
  - Conține 159 000 cuvinte, $[log_2 159\ 000] = 18\ Bits$

  => putem reprezenta fiecare cuvânt printr-un număr reprezentat pe 18 biți

  dar,
    - sunt prea mulți biți pe cuvânt
    - toată lumea are nevoie de un dicționar pentru decodate
    - merge doar ~~pentru limba engleză~~
  - Soluții:
    - Crearea adaptivă a dicționarului
    - LZW – introduce ideea că doar un dicționar inițial trebuie transmis
      - atât codorul cât și decodorul sunt capabili să creeze restul dicționarului
- LZW original,
  - folosea dicționare de 4K (valori/entries), primele 256 (0-255) fiind ASCII codes.

- LZW – înlocuiește șiruri de caractere printr-un cod
  - Nu face nici o analiză a textului/datelor de la intrare
  - doar adaugă fiecare șir nou de caractere de la intrare într-un tabel/dictionar
  - Coduri de 12 biti 0-256

```
STRING = get input character
WHILE there are still input characters DO
    CHARACTER = get input character
    IF STRING+CHARACTER is in the string table then
        STRING = STRING+character
    ELSE
        output the code for STRING
        add STRING+CHARACTER to the string table
        STRING = CHARACTER
    END of IF
END of WHILE
output the code for STRING
```

Problema 1

DPCM

Fie următoarea linie dintr-o imagine digitală pe nivele de gri:

$$\mathbf{u} = [100 \quad 110 \quad 120 \quad 128 \quad 64 \quad 64 \quad 64 \quad 64]$$

Dorim să codăm această linie de imagine (mai puţin primul pixel, $\boxed{u(0)=100}$) folosind modulaţia delta, având la dispoziţie un cuantizor pe 1 bit cu nivelele de cuantizare: {-15;15} şi funcţia de transfer:

$$\dot{e} = \begin{cases} -15, \text{daca } e < 0 \\ 15, \text{daca } e \geq 0 \end{cases}$$

u	e	$\dot{e}$	$\dot{u}$
100			85
110	(110-85)		

Se consideră reprezentarea valorilor de codat prin şirul de luminanţe $\{u(1),u(2),...,u(7)\} = \{110,120,...,64\}$. Dacă valoarea decodată reconstruită a primului eşantion, $\dot{u}(0) = \boxed{85}$ calculaţi:

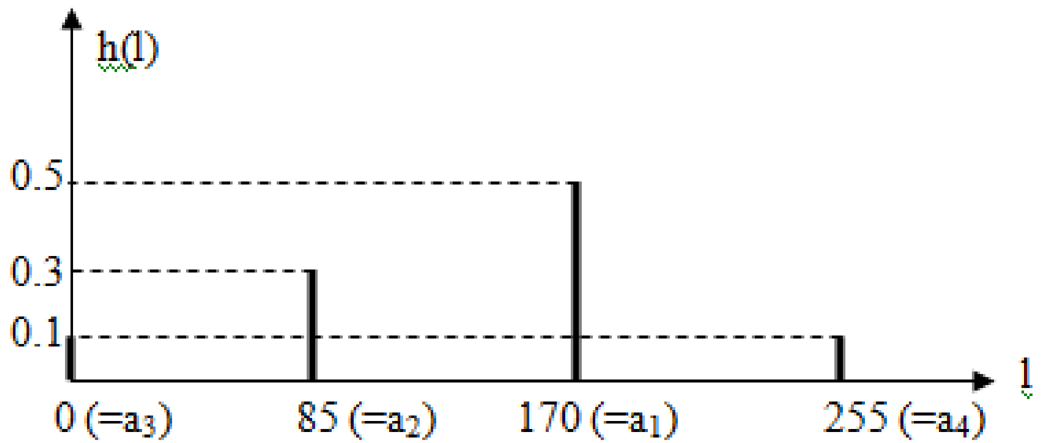
a) şirul erorilor de predicţie {e(n)}, n=1,2,...,7;

b) şirul erorilor de predicţie cuantizate $\left\{\dot{e}(n)\right\}$, n=1,2,...,7;

c) linia din imagine reconstruită la decodor.

Problema 2

Pentru o clasă de imagini, se aplică cuantizarea uniformă a nivelelor de gri pe 2 biţi, astfel încât în orice imagine din această clasă pot fi prezente doar nivelele de gri: 0; 85; 170; 255. Probabilităţile de apariţie ale celor 4 nivele de gri în imaginile din această clasă sunt date în histograma liniară normalizată a nivelelor de gri ale imaginilor din clasă din figură. Se va considera că cele 4 nivele de gri reprezintă simboluri posibile a fi emise de către o sursă S, în ordinea descrescătoare a probabilităţilor lor de apariţie: $S=\{a_1, a_2, a_3, a_4\}=\{170, 85, 0, 255\}$. Dacă pentru un bloc de imagine $\mathbf{U}[8\times4]$ din clasa menţionată, prima linie din bloc este: $\mathbf{u}(0)=[170 \quad 170 \quad 0 \quad 0]$, să se codeze această linie ca secvenţă emisă de sursa S folosind codarea aritmetică.

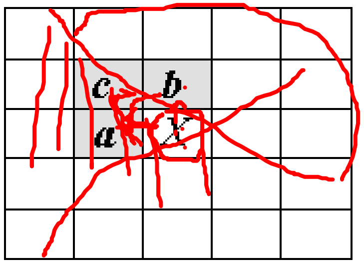


JPEG - Cele doua metode de compresie

- **Metoda de compresie cu pierderi bazată pe transformata cosinus discretă**
  - este vorba de formatul JPEG „clasic”, care permite rate de compresie importante (de 10:1 până la 20:1) păstrând în acelaşi timp o foarte bună calitate a imaginii; această metodă de compresie este ireversibilă.
- **Metoda de compresie predictivă fără pierderi**
  - nu au loc pierderi de informaţie şi este în consecinţă posibilă reproducerea exactă a imaginii originale, dar rata compresiei posibilă cu această metodă este mult mai modestă (aproximativ 2:1); această metodă de compresie este reversibilă.

JPEG fara pierderi

- NU foloseste algoritmii de compresie prin transformari – ~~DCT~~
- Factor de compresie slab ✓
- algoritmul JPEG fara pierderi:
  - codare diferentiala
  - codor Huffman sau aritmetic
- predictorii:
  - Eroarea de predicţie: $e=y-X$



$y = 0$

$y = a$

$y = b$

$y = c$

$y = a + b - c$

$y = a + \frac{b - c}{2}$

$y = b + \frac{a - c}{2}$

$y = \frac{a + b}{2}$

JPEG fara pierderi – codarea

- perechi de cuvinte – *categorie, amplitudine*
- *categoria* = numarul de biti necesari pentru codarea aplitudinii
- categoria se codeaza Huffman
- daca amplitudinea este pozitiva – valoarea amplitudinii
- daca amplitudinea este negativa – complementul valorii absolute

Exemplu de codare

- a=100, b=156, c=76, X=173
- $$e = \frac{a+b}{2} - X$$
- e = -45
- categoria 6
- rep. binara a lui 45 este 101101
- complementul lui este 010010
- –45 se reprezinta ca (6, 010010)
- daca codul Huffman a lui 6 este 1110
- –45 se codeaza cu 1110010010 (10 biti)

Categorie	Eroare de predicţie
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
11	-2047,...,-1024,1024, ..., 2047
12	-4095,...,-2048,2048, ..., 4095
13	-8191,...,-4096,4096, ..., 8191
14	-16383,...,-8192,8192, ..., 16383
15	-32767,...,-16384,16384, ..., 32767
16	32768

Compresie cu/fără pierderi