Universitatea "Politehnica" din Bucureşti
Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei

Tehnici Avansate de Prelucrarea şi Analiza Imaginilor

Curs 6 – Filtrarea neliniară

Ş.I. Bogdan IONESCU
Prof. Constantin VERTAN
Conf. Mihai CIUC

Plan Curs 6 – Filtrare neliniară

6.1. Filtrarea intrinsec neliniară

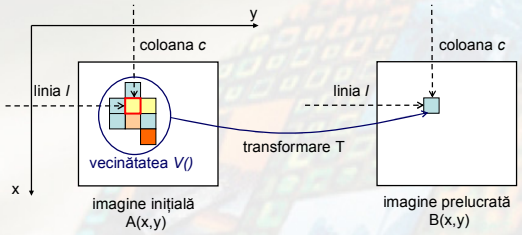
6.2. Filtrare mediană vectorială

6.3. Filtrarea neliniară adaptivă

6.4. Evaluarea calităţii filtrării

6.1. Filtrarea intrinsec neliniară

Definiţie operaţii de vecinătate



$$B(l,c) = T(A(V_{(l,c)}))$$

unde *A()* = imaginea iniţială, *B()* = imaginea modificată şi $V_{(l,c)}$ reprezintă o vecinătate a pixelului de coordonate *(l,c)*

Definiţie operaţii de vecinătate

transformarea este specificată integral dacă:
- se specifică vecinătatea $V_{(l,c)}$ (fereastră de prelucrare)
- se specifică modul de combinare al pixelilor vecini, funcţia *T()*.

→ în funcţie de *T()*, operaţiile de filtrare (de vecinătate) sunt:

↳ **filtrări liniare**: *T()* este o combinaţie liniară a valorilor pixelilor din vecinătate.

↳ **filtrări neliniare**: *T()* este o funcţie neliniară:

- *intrinsec neliniară* (filtrare neliniară),

- neliniară ca efect al adaptării la conţinut (filtrare adaptivă),

$$B(l,c) = T(A(l+m_1, c+n_1), \ldots, A(l+m_K, c+n_K))$$

Transformări intrinsec neliniare

$$B(l,c) = T(A(l+m_1, c+n_1), \ldots, A(l+m_K, c+n_K))$$

↳ funcţia *T()* este neliniară, exemple: min, max, log, exp, putere, etc.

→ ca şi în cazul filtrării liniare se doreşte:

- creşterea uniformităţii în interiorul regiunilor sau **filtrare de netezire**

- creşterea contrastului pe frontierele regiunilor sau **filtrare de contrastare.**

- creşterea uniformităţii în interiorul regiunilor ~ reducerea micilor variaţii ale valorilor pixelilor, datorate, de exemplu, zgomotului.

Zgomotul impulsiv

- un al tip de zgomot frevent în imagini (vezi şi *zgomotul alb Gaussian aditiv* din cursul anterior) îl reprezintă *zgomotul impulsiv*.

- acesta înlocuieşte aleator anumite valori ale pixelilor din imagine cu valorile extreme, în cazul nivelurilor de gri: alb (L-1) şi negru (0).

- efectul constă în împrăştierea unor puncte albe şi negre = sare şi piper ("salt and pepper").

$$A(l,c)=\begin{cases}0 & p/2\\ L-1 & p/2\\ A_0(l,c) & 1-p\end{cases}$$

unde *p* reprezintă probabilitatea zgomotului

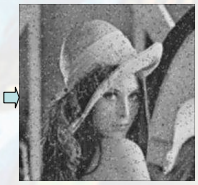
imagine perturbată, p=0.05

Zgomotul impulsiv

- încercăm eliminarea zgomotului de tip impulsiv folosind o filtrare de netezire liniară (mediere):

imagine neperturbată $A_0(l,c)$

imagine perturbată $A(l,c)$



filtrare de mediere

- filtrarea de netezire liniară amplifică efectul zgomotului prin împrăştierea acetuia (smearing),
→este nevoie de o operație de netezire neliniară

Filtrarea neliniară de ordonare

- este operația de ordonare neliniară ?

→ transformare liniară înseamnă
(se verifică principiul superpoziției):

$$T(\alpha\cdot A+\beta\cdot B)=\alpha\cdot T(A)+\beta\cdot T(B)$$

unde *A()* şi *B()* sunt două imagini, α şi β două constante iar *T()* este funcția transformării

- exemple:

A = (2,1,3) *B* = (1,3,2) $\alpha=\beta=1$

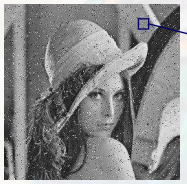
T(A) = (1,2,3) *T(B)* = (1,2,3)

$\alpha A+\beta B$ = (3,4,5) → *T(A)+T(B)* = (2,4,6) ≠ *T(A+B)* = (3,4,5)

Filtrarea neliniară de ordonare

- cum se poate folosi funcția de ordonare pentru a filtra zgomotul impulsiv ?

- acesta constă în valori extreme (0 şi L-1), astfel că pentru o anumită vecinătate tot ce trebuie să facem este să alegem o altă valoare cât mai departe de extreme,
→ ordonarea permite identificarea acestor pixeli



imagine perturbată

100 255 120
0 157 128
145 0 145

zgomot impulsiv

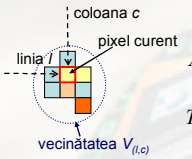
ordonare ↗

0, 0, 100, 120, 128, 145, 145, 157, 255

impulsurile se strâng spre capete

Filtrarea neliniară de ordonare

- generalizăm pentru o vecinătate $V_{(l,c)}$



$$A(V_{(l,c)})=\{A_1,A_2,...,A_K\}$$

reprezintă valorile pixelilor din fereastra de filtrare (*V*)

$$T(A(V_{(l,c)}))=\{A_{(1)},\le A_{(2)},\le,...,\le A_{(K)}\}$$

reprezintă valorile pixelilor ordonate crescător

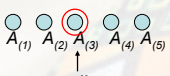
1 ≠ (1) = poziție în vector ordonat

- $A_{(1)}$ este valoarea minimă,
- $A_{(K)}$ este valoarea maximă,
- $A_{(i)}$ se numeşte statistica de ordine de "ordin *i*",

Filtrarea mediană

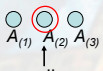
- ieşirea filtrului median este dată de valoarea din centrul vecinătății ordonate crescător = *statistică mediană*

K=5



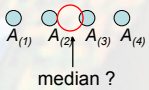
median

K=3



median

K=4



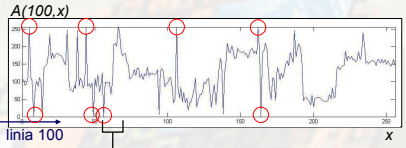
median ?

$$B_{median}(l,c)=\begin{cases}A_{\left(\frac{K+1}{2}\right)} & K\ impar\\ \frac{1}{2}\cdot A_{\left(\frac{K}{2}\right)}+\frac{1}{2}\cdot A_{\left(\frac{K}{2}+1\right)} & K\ par\end{cases}$$

Filtrarea mediană

- exemplu, K=3:

imagine perturbată A()



x=[55;68]

[126 0 99 96 96 99 64 102 206 209 202 255 244]

zgomot zgomot

Obs.: valorile alese corespund unei tranziții de contur (valorile cresc)

Filtrarea mediană

- exemplu, K=3 (continuare):

1: [126 0 99] 96 96 99 64 102 206 209 202 255 244]

[126 0 99] → [0 99 126] → median = 99

2: [126 0 99 96] 96 99 64 102 206 209 202 255 244]

[0 99 96] → [0 96 99] → median = 96

3: [126 0 99 96 96] 99 64 102 206 209 202 255 244]

[99 96 96] → [96 96 99] → median = 96

Filtrarea mediană

- exemplu, K=3 (continuare):

4: [126 0 99 96 96 99] 64 102 206 209 202 255 244]

[96 96 99] → [96 96 99] → median = 96

5: [126 0 99 96 96 99 64] 102 206 209 202 255 244]

[96 99 64] → [64 96 99] → median = 96

...

11:[126 0 99 96 96 99 64 102 206 209 202 255 244]

median = 244 ← [202 244 255] ← [202 255 244]

Filtrarea mediană

- exemplu, K=3 (continuare):

inițial

[126 0 99 96 96 99 64 102 206 209 202 255 244]

zgomot zgomot

filtrat

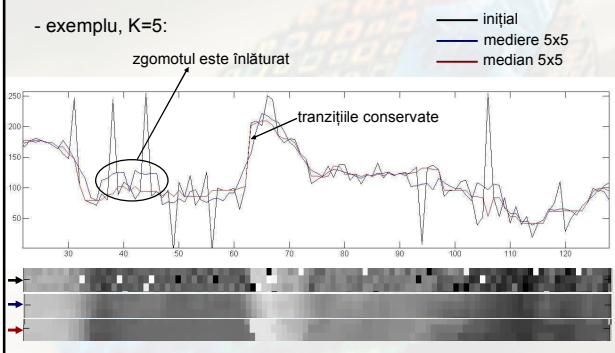
[99 99 96 96 96 96 99 102 206 206 209 244 244]

- valorile zgomotului impulsiv au fost transformate în valori ale pixelilor vecini,

- se observă păstrarea tranziției progresive a valorilor (crescătoare) aceasta fiind netezită de valori extreme (spikes).

Filtrarea mediană

- exemplu, K=5:



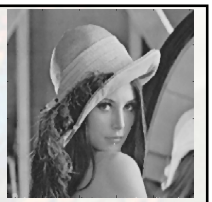
secțiunea transversală a liniei 100 din imagine (oX coordonate pixeli, oY nivel de gri)

Filtrarea mediană

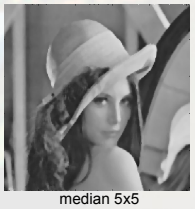
- exemple practice:

imagine inițială

zgomot impulsiv p=0.05



median 3x3



median 5x5

Filtrarea mediană

- exemple practice (continuare):

imagine inițială

zgomot impulsiv
p=0.1

median 3x3

median 5x5

Filtrarea mediană

- exemple practice (continuare):

imagine inițială

zgomot impulsiv
p=0.03

median 2x2

Filtrarea mediană - proprietăți

- filtrul median nu este un filtru liniar ≠ filtru de mediere,

- filtrul median este optimal pentru eliminarea zgomotului de tip impulsiv ("salt and pepper"),

- filtrul median comută cu orice funcție monotonă aplicată valorilor prelucrate:

$$median\{g(A_1), g(A_2), ..., g(A_K)\} = g(median\{A_1, A_2, ..., A_K\})$$

unde g() este o funcție monotonă iar $A_1,...,A_k$ reprezintă valorile prelucrate.

- filtrul median admite ceea ce numim *semnale rădăcină* = semnale ce nu sunt modificate prin filtrare (semnalele rădăcină ale unui filtru median de dimensiune *K* sunt secvențe monotone de dimensiune cel puțin *K*),

Filtrarea mediană - proprietăți

- filtrul median admite ceea ce numim *semnale rădăcină* = semnale ce nu sunt modificate prin filtrare (continuare)

→ porțiunile monotone ale semnalului nu sunt modificate (ex. platouri constante, tranziții progresive, etc.)

inițial: [2 3 5 7 9 12] median x3: [3 5 7 9]

inițial: [2 3 3 3 3 4] median x3: [3 3 3 3]

→ un semnal rădăcină se poate obține prin filtrare mediană repetitivă:

inițial: [10 3 2 11 1 0]

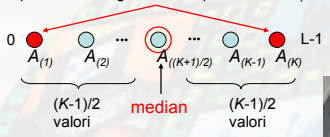
1: median x3: [10 3 3 2 1 0]

2: median x3: [10 3 3 2 1 0]

Filtrarea mediană - proprietăți

- *străpungerea filtrului median* = trecerea unui impuls de zgomot prin filtru

impulsurile se regăsesc la capetele secvenței



→ când apare pericolul ca zgomotul să treacă prin filtru ? când în vecinătatea considerată există cel puțin *(K-1)/2* valori zgomot de acelaşi tip (max sau min)

median 2x2
(inițial p=0.03)

Filtrarea mediană - proprietăți

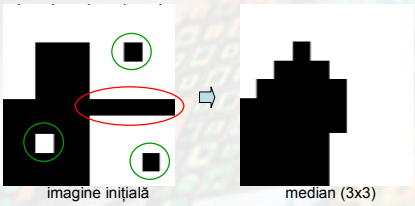
- dacă fereastra de filtrare este de lungime impară (există valoare de mijloc) atunci ieşirea filtrului median este întotdeauna o valoare din imagine,

median 3x3

- imaginea filtrată conține doar valori din imaginea inițială

Filtrarea mediană - dezavantaje

[C.Achard, J.Devars, M.Milgram]



imagine inițială — median (3x3)

- înlătură detaliile fine din imagine,
- înlătură regiunile izolate de puncte (nu face diferența dintre zgomot şi informație utilă),
- are o complexitate de calcul ridicată datorită sortării.

NB: exemplu a fost ales special pentru a accentua aceste limitări, în practică nu se filtrează median imagini binare.

Extinderea noțiunii de filtrare mediană

↳ filtrul median separabil

- prelucrarea 2D este înlocuită cu prelucrări 1D succesive (filtrul este descompus pe două direcții ortogonale):

100	255	120
0	157	128
145	0	145

→ [0 0 100 120 128 145 145 157 255]

median = 128

→ [100 120 255] median = 120
[0 128 157] median = 128
[0 145 145] median = 145

+ [120; 128; 145] median = 128

Extinderea noțiunii de filtrare mediană

↳ filtre de ordine

- generalizarea noțiunii de filtrare de ordonare,

- ieşirea unui filtru de ordine de ordin *j* (rank-order filter) este dată de statistica de ordine de ordin *j* a valorilor din vecinătatea considerată (fereastra de prelucrare):

$$rank_j\{A_1, A_2, ..., A_K\} = A_{(j)},\ j = 1, ..., K$$

unde $A_1,...,A_K$ reprezintă valorile prelucrate, *K* reprezintă numărul de valori din vecinătate, *j* este ordinul filtrului de ordine iar *(j)* reprezintă poziția în vectorul ordonat crescător.

Extinderea noțiunii de filtrare mediană

↳ filtre de ordine (continuare)

$$rank_j\{A_1, A_2, ..., A_K\} = A_{(j)},\ j = 1, ..., K$$

- dacă *j=(K+1)/2* (K=impar) → filtrare mediană
- dacă *j=1* → filtru min
- dacă *j=K* → filtru max

> proprietăți:

$$rank_j\{x_1 + y_1, ..., x_K + y_K\} \neq rank_j\{x_1, ..., x_K\} + rank_j\{y_1, ..., y_K\}$$

→ neliniar

$$rank_j\{g(x_1), ..., g(x_K)\} = g(rank_j\{x_1, ..., x_K\})$$

→ comută cu funcții monotone *g()*

Extinderea noțiunii de filtrare mediană

↳ filtre de ordine (continuare)

> proprietăți (continuare):

$$rank_j\{\alpha \cdot x_1 + \beta, ..., \alpha \cdot x_K + \beta\} = \alpha \cdot rank_j\{x_1, ..., x_K\} + \beta$$

→ comută cu operațiile liniare

- exemple:

imagine inițială — $rank_1$ = min (3x3) — $rank_3$ (3x3)

Extinderea noțiunii de filtrare mediană

↳ filtre de ordine (continuare)

- exemple (continuare):

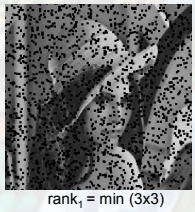
imagine inițială — $rank_5$ = median (3x3) — $rank_7$ (3x3) — $rank_9$ = max (3x3)

Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine (continuare)**

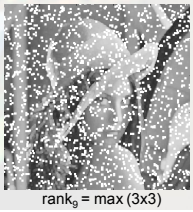
- acum şi cu zgomot mpulsiv, care este efectul ?

imagine cu zgomot impulsiv (p=0.05)



$rank_1$ = min (3x3)

$rank_5$ (3x3)

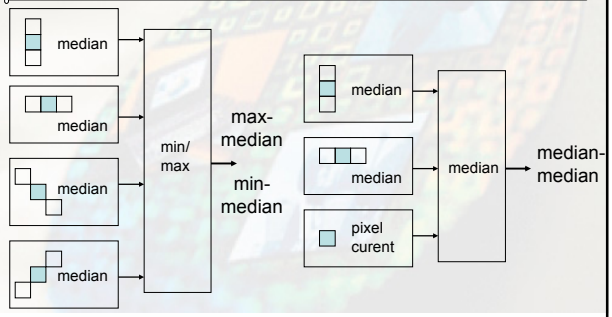


$rank_9$ = max (3x3)

Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine multietaj**

- multietaj = o succesiune de filtre de ordine de diferite ranguri.



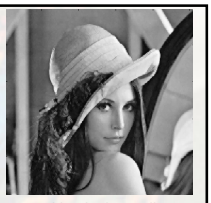
Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine multietaj (continuare)**

- exemple:

imagine inițială

multietaj max-median



multietaj min-median

multietaj median-median

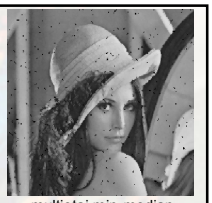
Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine multietaj (continuare)**

- exemple şi cu zgomot impulsiv:

imagine cu zgomot impulsiv (p=0.05)

multietaj max-median



multietaj min-median

multietaj median-median

Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine ponderate**

- filtrarea liniară, prin specificarea coeficienților de pondere ai filtrului, permite o flexibilitate/adaptabilitate ridicată (se pot imagina o infinitate de filtre),

- în cazul filtrării de ordine, singurul parametru de reglaj este ordinul statisticii utilizate (*j*), astfel se doreşte un control mai fin al filtrului = *ponderare*.

- ponderare = mărirea influenței relative a anumitor pixeli din fereastra de filtrare la calculul noii valori a pixelului curent (ex. pixelul central să aibă o pondere mai importantă asupra rezultatului filtrării decât pixelii vecini)

- în cazul liniar, ponderarea se făcea prin înmulțire cu scalari, se poate aplica acelaşi procedeu în cazul filtrelor de ordine ?

Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine ponderate (continuare)**

- filtrarea de ordine = ordonare → a da o pondere mai importantă înseamnă *a repeta valoarea de mai multe ori* mărind probabilitatea de a se regăsi în statistica aleasă.

- mască de ponderare:

$$W = \begin{bmatrix} w_{-1,-1} & w_{-1,0} & w_{-1,1} \\ w_{0,-1} & \boxed{w_{0,0}} & w_{0,1} \\ w_{1,-1} & w_{1,0} & w_{1,1} \end{bmatrix}$$

unde w_{ij} reprezintă numărul de repetiții ale valorilor pixelului *A(l+i,c+j)* (ponderi, pozitive şi ≠ 0) şi *(l,c)* sunt coordonatele pixelului curent prelucrat.

- mulțimea de valori rezultată din multiplicarea valorilor din fereastra de prelucrare pe baza măştii de ponderare = multiset.

Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine ponderate (continuare)**

- exemplu, să considerăm că filtrăm pixelul curent următor:

coloana *c*

linia *l* ----->

$$\begin{matrix} 1 & 3 & 3 \\ 2 & (2) & 1 \\ 4 & 3 & 5 \end{matrix} \qquad W = \begin{bmatrix} 1 & 2 & 1 \\ 2 & (3) & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

fără ponderare: [1 1 2 2 (3) 3 3 4 5] median = 3

cu ponderare (multiset):

multiset

[1 3 3 3 2 2 2 2 2 1 1 4 3 3 5]

multiset (ordonat)

[1 1 1 2 2 2 2 (2) 3 3 3 3 3 4 5]

median = 2 (egală/apropiată de pixelul filtrat, pondere mai mare)

Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine ponderate (continuare)**

filtrarea de ordine ponderată se poate scrie:

$$rank_{\ j}\left\{A(V_{(l,c)}) \Diamond W\right\} = A_{(j)}$$

unde $V_{(l,c)}$ reprezintă vecinătatea pixelului curent (l,c), W este masca de ponderare iar $A_{(j)}$ este statistica de ordin j din multisetul rezultat.

→ numărul de elemente = suma ponderilor $\quad Card\left\{A(V_{(l,c)}) \Diamond W\right\} = \sum_{k=1}^{K} w_k$

→ fără ponderare: $\quad daca\ \ w_k = 1, k = 1,..., K$

→ ordinul statisticii mediane ponderate: $\quad \frac{1}{2} \cdot \left(\sum_{k=1}^{K} w_k + 1 \right)$

Extinderea noțiunii de filtrare mediană

↳ **filtre de ordine ponderate (continuare)**

- caz particular, filtrul de ordine central ponderat:

$$W = \begin{bmatrix} 1 & 1 & 1 \\ 1 & (p) & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

pixelul curent prelucrat are ponderea cea mai mare (p>1)

- exemple:

median clasic (3x3)

zg. impulsiv (p=0.05)

median ponderat (3x3)

median ponderat (3x3)

L-filtre

-şi mai general, oferă o mai bună flexibilitate operațiilor de filtrare neliniară bazate pe ordonare prin apropierea de modul de specificare al filtrelor liniare:

$$Lfilter\ \left\{A_1, A_2, ..., A_K\right\} = \sum_{i=1}^{K} w_i A_{(i)}$$

unde $A_1,...,A_K$ reprezintă valorile prelucrate, K reprezintă numărul de valori din vecinătate, $A_{(i)}$ reprezintă statistica de ordine de ordin i iar w_i ponderea acesteia la calculul ieşirii filtrului (nu are legătură cu ponderile anterioare).

-cazuri particulare:

→ filtrul de ordine de ordin j: $\quad w_i = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases}$

$$Lfilter\ \left\{A_1, A_2, ..., A_K\right\} = A_{(j)}$$

L-filtre

$$Lfilter\ \left\{A_1, A_2, ..., A_K\right\} = \sum_{i=1}^{K} w_i A_{(i)}$$

-cazuri particulare (continuare):

→ filtrul de mediere: $\quad w_i = \frac{1}{K}$

$$Lfilter\ \left\{A_1, A_2, ..., A_K\right\} = \frac{1}{K} \sum_{i=1}^{K} A_{(i)} = \frac{1}{K} \sum_{k=1}^{K} A_k$$

→ filtrul de mijloc: $\quad w_1 = w_K = \frac{1}{2}, w_i = 0 \text{ pentru } i \notin \{1, K\}$

$$Lfilter\ \left\{A_1, A_2, ..., A_K\right\} = \frac{A_{(1)} + A_{(K)}}{2} = \frac{\min + \max}{2}$$

L-filtre

$$Lfilter\ \left\{A_1, A_2, ..., A_K\right\} = \sum_{i=1}^{K} w_i A_{(i)}$$

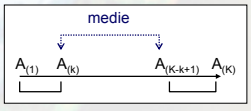
-cazuri particulare (continuare):

→ filtrul de cvasi-mijloc:

$$w_k = w_{K+1-k} = \frac{1}{2}, w_i = 0 \text{ pentru } i \notin \{k, K+1-k\}$$

unde k specifică poziția relativă a mijlocului (mediei).

$$Lfilter\ \left\{A_1, A_2, ..., A_K\right\} = \frac{A_{(k)} + A_{(K-k+1)}}{2}$$



L-filtre

$$Lfilter\ \{A_1, A_2, ..., A_K\} = \sum_{i=1}^{K} w_i A_{(i)}$$

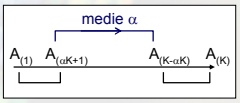
-cazuri particulare (continuare):

→ medie α-reglabilă:

$$w_k = \begin{cases} \dfrac{1}{K \cdot (1-2\cdot\alpha)} & k \in [\alpha \cdot K + 1; K - \alpha \cdot K] \\ 0 & k \notin [\alpha \cdot K + 1; K - \alpha \cdot K] \end{cases}$$

unde α reglează calculul mediei.

$$Lfilter\ \{A_1, A_2, ..., A_K\} = \frac{1}{K \cdot (1-2\cdot\alpha)} \sum_{i=\alpha K+1}^{K-\alpha K} A_{(i)}$$



L-filtre

$$Lfilter\ \{A_1, A_2, ..., A_K\} = \sum_{i=1}^{K} w_i A_{(i)}$$

-se observă faptul că ieşirea filtrului se calculează pe baza ponderării valorilor pixelilor din vecinătatea considerată

→ sunt valabile considerațiile de la filtrarea liniară, astfel L-filtrele sunt de două tipuri:

- **L-filtre de netezire** (~ FTJ) = creşterea uniformității în interiorul regiunilor, $\sum_{i=1}^{K} w_i = 1$
- **L-filtre de derivare** (~ FTS) = punerea în evidență a tranzițiilor, $\sum_{i=1}^{K} w_i = 0$

L-filtre

- din ce categorie fac parte filtrele enumerate anterior ?

$$Lfilter\ \{A_1, A_2, ..., A_K\} = A_{(j)}$$

$$Lfilter\ \{A_1, A_2, ..., A_K\} = \frac{1}{K}\sum_{i=1}^{K} A_{(i)} = \frac{1}{K}\sum_{k=1}^{K} A_k$$

$$Lfilter\ \{A_1, A_2, ..., A_K\} = \frac{A_{(1)} + A_{(K)}}{2} = \frac{\min + \max}{2}$$

$$Lfilter\ \{A_1, A_2, ..., A_K\} = \frac{A_{(k)} + A_{(K-k+1)}}{2}$$

$$Lfilter\ \{A_1, A_2, ..., A_K\} = \frac{1}{K \cdot (1-2\cdot\alpha)} \sum_{i=\alpha K+1}^{K-\alpha K} A_{(i)}$$

filtre de netezire

L-filtre

- operațiile de filtrare au ca scop eliminarea perturbațiilor:

tip zgomot	L-filtru adaptat	ponderi *w*
impulsiv	median	$w_{\frac{K+1}{2}} = 1, w_i = 0\ i \neq \frac{K+1}{2}$
aditiv Gaussian	medie	$w_i = \frac{1}{K}$
impulsiv+ aditiv Gaussian	medie α reglabilă	$w_k = \begin{cases} \frac{1}{K\cdot(1-2\cdot\alpha)} & k \in [\alpha\cdot K+1; K-\alpha\cdot K] \\ 0 & rest \end{cases}$
uniform	mijloc	$w_1 = w_K = \frac{1}{2}, w_i = 0$ in rest
impulsiv+ uniform	cvasi-mijloc	$w_k = w_{K+1-k} = \frac{1}{2}, w_i = 0$ in rest

L-filtre

- exemple:

imagine inițială

L-filtru mijloc (3x3)

L-filtru medie α=1/3 (3x3)

L-filtru cvasi-mijloc, k=3 (3x3)

L-filtre

- exemple şi cu zgomot:

imagine cu zgomot impulsiv (p=0.04) + uniform (disp=5)

L-filtru mijloc (3x3)

L-filtru medie α=1/3 (3x3)

L-filtru cvasi-mijloc, k=4 (3x3)

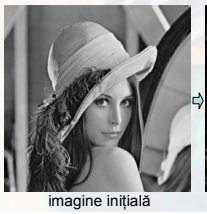
L-filtre

- gradientul morfologic (L-filtru de derivare):

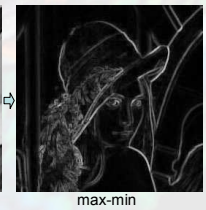
$$Lfilter\ \{A_1, A_2, ..., A_K\} = A_{(K)} - A_{(1)}$$
$$= \max - \min$$
$$\sum_{i=1}^{K} w_i = 1 - 1 = 0$$

- exemple:

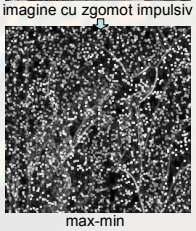
imagine cu zgomot impulsiv



imagine inițială



max-min



max-min

Filtre de ordonare de domeniu

- sau LUM (lower-upper-middle), introduc posibilitatea de adaptare a filtrelor de ordine, astfel:

- **filtrul LUM de netezire:**

$$LUM_j = \begin{cases} A_{(j)} & A^* < A_{(j)} \\ A_{(K-j+1)} & A^* > A_{(K-j+1)} \\ A^* & altfel \end{cases}$$

unde $A_1,...,A_K$ reprezintă valorile prelucrate, $A_{(j)}$ reprezintă statistica de ordine de ordin j, A^* este valoare pixelului curent iar j este un parametru de reglaj, $j \in [1;(K+1)/2]$.

- dacă j=1 → $LUM_1 = A^*$ (filtru trece tot)

- dacă j=(K+1)/2 → $LUM_{med} = A_{(j)}$ (filtru median)

Filtre de ordonare de domeniu

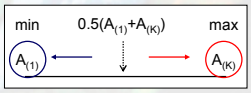
- sau LUM (lower-upper-middle) introduc posibilitatea de adaptare a filtrelor de ordine (continuare):

- **filtrul LUM de conturare:**

$$LUM_j = \begin{cases} A_{(j)} & A_{(j)} < A^* < 0.5 \cdot (A_{(j)} + A_{(K-j+1)}) \\ A_{(K-j+1)} & 0.5 \cdot (A_{(j)} + A_{(K-j+1)}) < A^* < A_{(K-j+1)} \\ A^* & altfel \end{cases}$$

- dacă j=1 →

LUM$_1$= filtru de reliefare extremă



- dacă j=(K+1)/2 → $LUM_{TT} = A^*$ (filtru trece tot)

Filtre de ordonare de domeniu

- exemple LUM conturare:

$$LUM_j = \begin{cases} A_{(j)} & A_{(j)} < A^* < 0.5 \cdot (A_{(j)} + A_{(K-j+1)}) \\ A_{(K-j+1)} & 0.5 \cdot (A_{(j)} + A_{(K-j+1)}) < A^* < A_{(K-j+1)} \\ A^* & altfel \end{cases} \quad j \in [1;(K+1)/2].$$

imagine inițială

accentuare j=3 (3x3)

accentuare j=1 (3x3)

Filtre de ordonare de domeniu

- exemple LUM netezire:

$$LUM_j = \begin{cases} A_{(j)} & A^* < A_{(j)} \\ A_{(K-j+1)} & A^* > A_{(K-j+1)} \\ A^* & altfel \end{cases} \quad j \in [1;(K+1)/2].$$

imagine cu zgomot impulsiv

netezire j=3 (3x3)

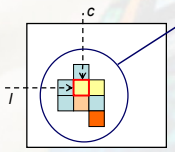
netezire j=5 ??? (3x3)

6.2. Filtrare mediană vectorială

Filtrarea mediană vectorială

- dacă filtrarea liniară de vecinătate poate fi aplicată în cazul imaginilor color pe baza strategiilor prezentate în cursul 2, nu putem spune acelaşi lucru şi despre filtrarea de ordonare, şi în particular despre filtrarea mediană,

→ care este relația de ordine dintre două culori ?



imagine color A(x,y)

$$V_{(l,c)} = \{(l+m_1, c+n_1), ..., (l+m_K, c+n_K)\}$$

$$A(V_{(l,c)}) = \{A(l+m_1, c+n_1), ..., A(l+m_K, c+n_K)\}$$

$A(l+m_k, c+n_k)$ sunt vectori: $A_R(l+m_k, c+n_k)$, $A_G(l+m_k, c+n_k)$, $A_B(l+m_k, c+n_k)$

Filtrarea mediană vectorială

- este evident faptul că a ordona după fiecare componentă, independent, are ca rezultat amestecarea culorilor.

- *punerea problemei*: având la dispoziție K vectori (valorile pixelilor din fereastra de analiză), cum putem determina statistica mediană:

$$median\{A_1, A_2, ..., A_K\} = ?$$ unde A_k sunt vectori de culoare, $k=1,...,K$, ex. (A_k^R, A_k^G, A_k^B)

→ construim o relație de ordine:

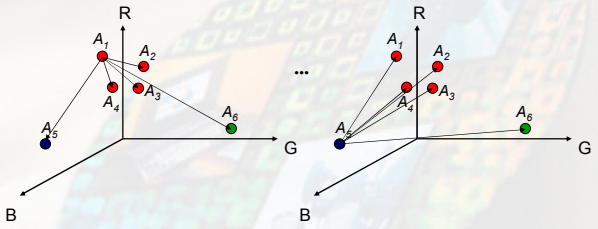
↳ **filtrare mediană de distanță (median vectorial)**

$$D_k = \sum_{i=1}^{K} \| A_k - A_i \|, k = 1, ..., K$$

$$D_k^E = \sum_{i=1}^{K} \sqrt{(A_k^R - A_i^R)^2 + (A_k^G - A_i^G)^2 + (A_k^B - A_i^B)^2}$$

Filtrarea mediană vectorială

↳ **filtrare mediană de distanță (continuare)**

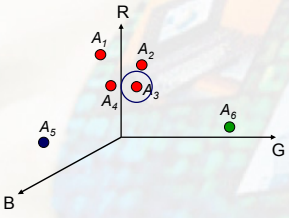


$$\{A_1 \quad A_2 \quad A_3 \quad A_4 \quad ... \quad A_K\}$$
$$\{D_1 \quad D_2 \quad D_3 \quad D_4 \quad ... \quad D_K\}$$

Filtrarea mediană vectorială

↳ **filtrare mediană de distanță (continuare)**

$$median\{A_1, A_2, ..., A_K\} = A_M \,|_{D_M = \min_{k=1,...,K}\{D_k\}}$$



- statistica mediană corespunde vectorului A_M ce furnizează distanța cumulată minimă

- median = valoarea cea mai "populară", distanță cumulată minimă ~ vectorul cel mai apropiat de ceilalți.

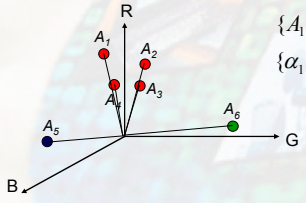
- ca şi în cazul medianului scalar nu apar valori noi în imagine.

Filtrarea mediană vectorială

↳ **filtrare mediană direcțională**

- de această dată, relația de ordine este dată de o sumă de direcții (unghiuri), astfel:

$$\alpha_k = \sum_{i=1}^{K} \angle(A_k, A_i)$$ unde $\angle(A_k, A_i)$ returnează unghiul dintre vectorii A_k şi A_i relativ la o origine (ex. originea sistemului)



$$\{A_1 \quad A_2 \quad A_3 \quad A_4 \quad ... \quad A_K\}$$
$$\{\alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad ... \quad \alpha_K\}$$

$$median\{A_1, A_2, ..., A_K\} = A_M \,|_{\alpha_M = \min_{k=1,...,K}\{\alpha_k\}}$$

Filtrarea mediană vectorială

↳ **filtrare mediană direcțională de distanță**

- se foloseşte atât distanța dintre vectori cât şi unghiul:

$$\Omega_k = D_k^{1-w} \cdot \alpha_k^w, \; k = 1, ..., K$$ unde $w \in [0;1]$ *este un parametru de reglaj ce modulează influența relativă a celor doi termeni*

$$D_k = \sum_{i=1}^{K} \| A_k - A_i \|, \quad \alpha_k = \sum_{i=1}^{K} \angle(A_k, A_i)$$

$$\Omega_k = \left(\sum_{i=1}^{K} \| A_k - A_i \| \right)^{1-w} \cdot \left(\sum_{j=1}^{K} \angle(A_k, A_j) \right)^{w}$$

Filtrarea mediană vectorială

↳ **filtrare mediană direcțională de distanță (continuare)**

$$\Omega_k = \left(\sum_{i=1}^{K} \| A_k - A_i \|\right)^{1-w} \cdot \left(\sum_{j=1}^{K} \angle(A_k, A_j)\right)^{w}$$

$$\{A_1 \quad A_2 \quad A_3 \quad A_4 \quad \dots \quad A_K\}$$
$$\{\Omega_1 \quad \Omega_2 \quad \Omega_3 \quad \Omega_4 \quad \dots \quad \Omega_K\}$$
$$\Rightarrow \quad median\{A_1, A_2, \dots, A_K\} = A_M \big|_{\Omega_M = \min_{k=1,\dots,K}\{\Omega_k\}}$$

↳ **filtrare mediană prin amestecul biților**

- o abordare diferită, inspirată de medianul scalar unde datele erau reprezentate pe 8 biți,

- în cazul scalar, fiecare pixel are asociată o secvență de biți, se va proceda similar şi pentru cazul vectorial,

Filtrarea mediană vectorială

↳ **filtrare mediană prin amestecul biților (continuare)**

- să considerăm cazul RGB:

$A_i = (A_i^R, A_i^G, A_i^B)$ unde $size(A_i) = 3 \cdot 8 = 24 biti$

$binar(A_i^R)$=$r_1 r_2 r_3 r_4 r_5 r_6 r_7 r_8$ (bitul cel mai semnificativ = indice 1)

$binar(A_i^G)$=$g_1 g_2 g_3 g_4 g_5 g_6 g_7 g_8$

$binar(A_i^B)$=$b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8$

$A_i \sim v_i$=[$r_1 g_1 b_1 r_2 g_2 b_2 r_3 g_3 b_3 r_4 g_4 b_4 r_5 g_5 b_5 r_6 g_6 b_6 r_7 g_7 b_7 r_8 g_8 b_8$]

bit-mixing

Filtrarea mediană vectorială

↳ **filtrare mediană prin amestecul biților (continuare)**

$$\{A_1 \quad A_2 \quad A_3 \quad A_4 \quad \dots \quad A_K\}$$
$$\{v_1 \quad v_2 \quad v_3 \quad v_4 \quad \dots \quad v_K\}$$

sortare ↗ în funcție de v

$$\{A_{(1)} \quad A_{(2)} \quad A_{(3)} \quad A_{(4)} \quad \dots \quad A_{(K)}\}$$
$$\{v_{(1)} \quad v_{(2)} \quad v_{(3)} \quad v_{(4)} \quad \dots \quad v_{(K)}\}$$

$median\{A_1, A_2, \dots, A_K\} = A_{\left(\frac{K+1}{2}\right)}$ şi anume pixelul ce corespunde vectorului v median, $v_{\left(\frac{K+1}{2}\right)}$

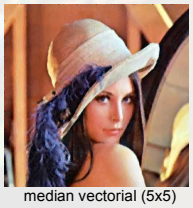
Filtrarea mediană vectorială

- exemple:

imagine inițială

imagine cu zgomot impulsiv (p=0.05)

median vectorial (3x3)



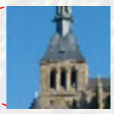
median vectorial (5x5)

[E.Dinet & F. Robert-Inacio]

Filtrarea mediană vectorială

- efectul asupra frontierelor (fereastră 3x3) :

imagine inițială



median vectorial

median direcțional

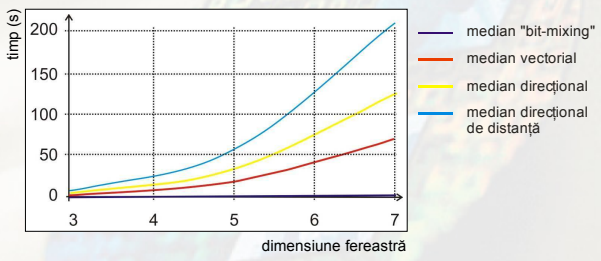
median direcțional de distanță

median "bit-mixing"

[E.Dinet & F. Robert-Inacio]

Filtrarea mediană vectorială

- calcul distanțe = complexitate de calcul ridicată:

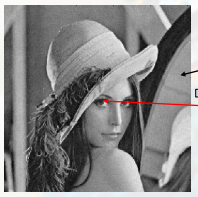


6.3. Filtrarea neliniară adaptivă

Filtrarea adaptivă

- o limitare a operațiilor de filtrare discutate anterior este dată de faptul că acestea se aplică *întregii imaginii* modificând mai mult sau mai puțin pixelii după aceeaşi "rețetă",

→ cele două situații de interes: contururile şi zonele uniforme sunt tratate de cele mai multe ori la fel:



imagine afectată de zgomot alb

mediere fereastră 5x5

- medierea a eliminat puternic zgomotul alb în zonele uniforme,
- dar, aplicată şi contururilor a dus la înceţoşarea imaginii

Filtrarea adaptivă

idee: filtrarea să fie realizată în funcție de conținutul imaginii.

adaptare = modificarea parametrilor de definiție ai unei prelucrări, în funcție de condițiile locale (vecinătatea pixelului analizat), pentru fiecare poziție/pixel din imaginea de prelucrat.

- nivelul de granularitate maxim ar fi obținut dacă adaptăm un filtru pentru fiecare pixel din imagine în parte.

- adaptarea impune existența unui mod de măsurare cantitativă, obiectivă, a efectelor de prelucrare (dorite sau nedorite) induse în imagine în funcție de parametrii ce definesc filtrele.

→ practic, adaptarea se obține de regulă în urma minimizării unor funcții de eroare/cost

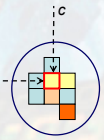
Filtrarea adaptivă

parametrii de relgaj ai unui filtru

→ filtrul dorit este o funcție de:

- forma vecinătății considerate $V_{(l,c)}$
- numărul de pixeli din vecinătate K
- ponderile filtrului ($w_{m,n}$): $B(l,c) = \sum w_{m,n} \cdot A(l+m, c+n)$

sau tipul funcției de transformare ($T()$): $B(l,c) = T(A(V_{(l,c)}))$



→ indiferent de modul de modificare al valorilor din fereastra de prelucrare (ponderi liniare, funcții neliniare,...) datorită adaptării prelucrarea rezultată este neliniară !

Filtrarea adaptivă

câteva tipuri de abordări:

→ **adaptarea formei** ferestrei de filtrare,
→ **adaptarea coeficienților** funcțiilor de combinare a valorilor,
→ **adaptarea filtrelor intrinsec neliniare**

↳ **adaptarea formei vecinătății**

- **filtrarea de netezire** este evaluată prin diferențele dintre imaginea filtrată şi cea inițială, dacă diferențele sunt mari → este foarte probabil ca netezirea să fi înceţoşat contururile din imagine.

ipoteze: zgomotul este mai slab decât contururile imaginii şi variațiile datorate zgomotului sunt mai mici decât variațiile dintre valorile regiunilor alăturate.

Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

- această abordare presupune un "comutator" ce permite alegerea pe baza unor parametri (praguri de acceptare a variațiilor) a valorii de ieşire cea mai adaptată (din mai multe valori posibile).

A. comutare mediere/trece-tot

$$B(l,c) = \begin{cases} \overline{A(V_{(l,c)})} & \left|A(l,c) - \overline{A(V_{(l,c)})}\right| < \tau \\ A(l,c) & altfel \end{cases}$$

τ → prelucrare de vecinătate

$A(l,c)$ → pixel curent

unde $A(x,y)$ este imaginea inițială afectată de zgomot, $B(x,y)$ este imaginea filtrată, (l,c) este pixelul curent iar τ este un prag (threshold)

Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

A. comutare mediere/trece-tot (continuare)

$$B(l,c)=\begin{cases}\overline{A(V_{(l,c)})} & \left|A(l,c)-\overline{A(V_{(l,c)})}\right|<\tau \\ A(l,c) & altfel\end{cases}$$

cu alte cuvinte: valoarea nouă a unui pixel = media aritmetică a valorilor din vecinatatea sa numai dacă această valoare nu este "prea diferită" de valoarea inițială.

$B(l,c)$ — $\overline{A(V_{(l,c)})}$ netezire

— $A(l,c)$ trece-tot

parametrul τ ?

Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

A. comutare mediere/trece-tot (continuare)

- exemple:

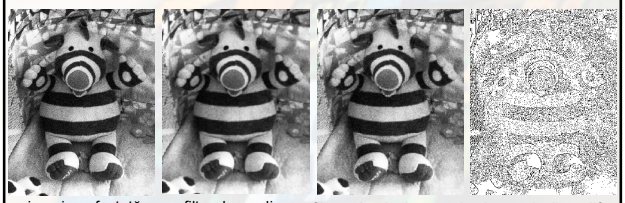
imagine afectată de zgomot alb | filtru de mediere (3x3) | filtru adaptiv de netezire (τ =13, 72% din pixelii sunt modif.)

Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

A. comutare mediere/trece-tot (continuare)

- exemple (continuare):



imagine afectată de zgomot | filtru de mediere (3x3) | filtru adaptiv de netezire (τ=13, 77% din pixelii sunt modifi.) | hartă pixeli modif. (alb=modif, negru=nemodif.)

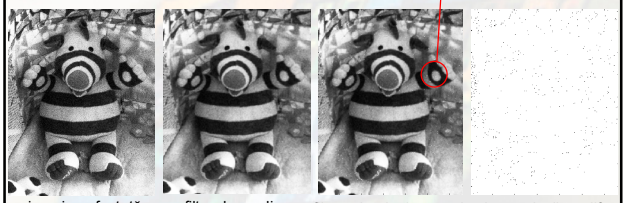
Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

A. comutare mediere/trece-tot (continuare)

- exemple (continuare):

nu am mediat valorile extreme (zgomot)



imagine afectată de zgomot | filtru de mediere (3x3) | filtru adaptiv de netezire (τ=60, 98% din pixelii sunt modif.) | hartă pixeli modif. (alb=modif, negru=nemodif.)

Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate

filtrarea se realizează de această dată orientat (după anumite direcții):

- exemplu: $V^1_{(l,c)}$ $V^2_{(l,c)}$ $V^3_{(l,c)}$ $V^4_{(l,c)}$

- dacă notăm: $B_i(l,c)=T_{\theta_i}(A(V^i_{(l,c)}))$ având direcția θ_i , $i=1,\ldots,4$

$$B(l,c)=B_k(l,c)\big|_{k=\arg\min_i\{|A(l,c)-B_i(l,c)|\}}$$

unde $A(x,y)$ este imaginea inițială afectată de zgomot, $B(x,y)$ este imaginea filtrată, (l,c) este pixelul curent.

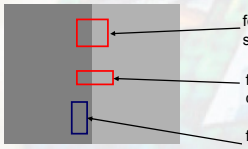
Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate (continuare)

$$B(l,c)=B_k(l,c)\big|_{k=\arg\min_i\{|A(l,c)-B_i(l,c)|\}}$$

cu alte cuvinte, ieşirea filtrului global este ieşirea filtrului de orientare θ_k pentru care diferența față de valoarea inițială este minimă = netezirea este cea mai "slabă".



fereastră de filtrare izotropă, strică conturul

fereastră de filtrare liniară perpendiculară pe contur, strică conturul

fereastră de filtrare liniară paralelă cu conturul, nu strică conturul

frontieră

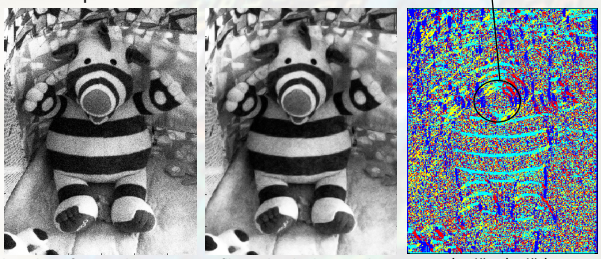
Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate (continuare)

- exemple:

se observă cele 4 direcții



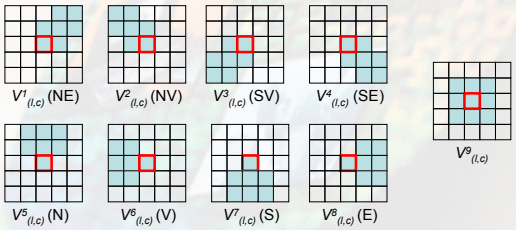
imagine afectată de zgomot | filtru adaptiv de netezire 4 vecinătăți | hartă orientări (4 direcții 4 culori)

Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate (continuare)

- un filtru clasic: filtrul Nagao, foloseşte următoarele tipuri de vecinătăți:



Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate (continuare)

- filtrul Nagao (continuare):

$$B_i(l,c) = T_i(A(V^i_{(l,c)})), i = 1,...,9$$

rezultatul filtrării după vecinătatea $A(V^i_{(l,c)}) = \{A^i_1, ..., A^i_K\}$

$$\sigma_i^2 = Var(A(V^i_{(l,c)})) = \frac{1}{K} \cdot \sum_{k=1}^{K} (A^i_k - \overline{A(V^i_{(l,c)})})^2$$

$$B(l,c) = B_k(l,c)\big|_{k=\arg\min_i \{\sigma_i^2\}}$$

cu alte cuvinte, ieşirea filtrului global este ieşirea filtrului orientat a cărui vecinătate ($A(V^i)$) are o varianță minimă (vecinătatea cea mai uniformă) = introduce diferențe minime

Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate (continuare)

- filtrul Nagao, exemple:

imagine afectată de zgomot | filtru Nagao de netezire (5x5) | filtru median (5x5) | filtru de mediere (5x5)

Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate (continuare)

- filtrul Nagao, exemple (continuare):

[ENST fr]

imagine inițială | filtru Nagao de netezire (9x9) | filtru median (7x7) | filtru de mediere (7x7)

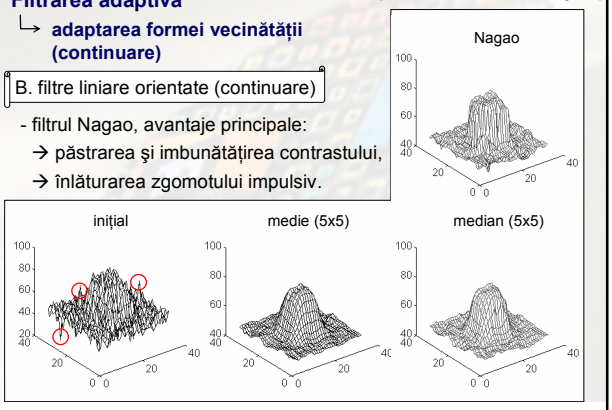
Filtrarea adaptivă

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate (continuare)

- filtrul Nagao, avantaje principale:
  - → păstrarea şi imbunătățirea contrastului,
  - → înlăturarea zgomotului impulsiv.

[C.Achard, J.Devars, M.Milgram]



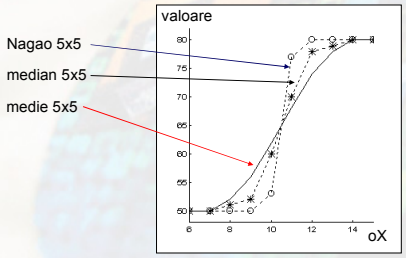
Filtrarea adaptivă

[C.Achard, J.Devars, M.Milgram]

↳ **adaptarea formei vecinătății (continuare)**

B. filtre liniare orientate (continuare)

- filtrul Nagao (continuare):



Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare

- pentru fiecare pixel prelucrat, coeficienții filtrului vor fi modificați în funcție de valorile imaginii (vecinătate),
→ teoretic pentru fiecare pixel avem un filtru diferit

- scopul principal este evitarea netezirii contururilor (efectului de blur),

- exemplu clasic: **filtrul LLMMSE** - Locally Linear Minimal Mean Squared Error (Lee)

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE (continuare)

principiu: imaginea filtrată se construieşte ca o combinație liniară convexă a imaginii originale (dar posibil degradate) şi a imaginii obținute prin medierea aritmetică în fiecare pixel a valorilor din imaginea originală:

$$B = \alpha \cdot \overline{A} + (1-\alpha) \cdot A$$

$$\overline{A}(l,c) = T_{mediere}(A(V_{(l,c)}))$$

unde $B()$ este imaginea filtrată, $A()$ este imaginea inițială, $\overline{A}()$ este imaginea mediată în fiecare pixel iar α un coeficient de reglaj.

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE (continuare)

$$B = \alpha \cdot \overline{A} + (1-\alpha) \cdot A$$

- dacă imaginea este perturbată de zgomot alb aditiv, atunci:

$A = A_0 + z$ unde A_0 este imaginea inițială fără zgomot.

$$\overline{z} = 0, \ \overline{A_0 \cdot z} = \overline{A_0} \cdot \overline{z} = 0$$

$$\Rightarrow B = \alpha \cdot \overline{(A_0 + z)} + (1-\alpha) \cdot (A_0 + z)$$

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE (continuare)

$$B = \alpha \cdot \overline{(A_0 + z)} + (1-\alpha) \cdot (A_0 + z)$$

$$B = \alpha \cdot \overline{A_0} + (1-\alpha) \cdot (A_0 + z)$$

- eroarea dintre $B()$ şi $A_0()$ se poate scrie astfel (ideal se caută ca în urma filtrării să obținem imaginea neperturbată):

$$\varepsilon = A_0 - B = A_0 - \alpha \cdot \overline{A_0} - (1-\alpha) \cdot (A_0 + z)$$

$$= \alpha \cdot (A_0 - \overline{A_0}) - (1-\alpha) \cdot z$$

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE (continuare)

$$\overline{\varepsilon^2} = \overline{\left(\alpha \cdot (A_0 - \overline{A_0}) - (1-\alpha) \cdot z\right)^2}$$

$$\overline{\varepsilon^2} = \alpha^2 \cdot \overline{(A_0 - \overline{A_0})^2} + (1-\alpha)^2 \cdot \overline{z^2} -$$

$$2 \cdot \alpha \cdot (1-\alpha) \cdot \overline{(A_0 - \overline{A_0}) \cdot z}$$ =0

$$\overline{\varepsilon^2} = \alpha^2 \cdot \sigma_{A_0}^2 + (1-\alpha)^2 \cdot \sigma_z^2$$

variante locale

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE (continuare)

$$\overline{\varepsilon^2} = \alpha^2 \cdot \sigma_{A_0}^2 + (1-\alpha)^2 \cdot \sigma_z^2$$

- căutăm să minimizăm eroarea pătratică medie funcție de parametrul α, astfel:

$$\frac{\partial \overline{\varepsilon^2}}{\partial \alpha} = 0 \Rightarrow \frac{\partial \overline{\varepsilon^2}}{\partial \alpha} = 2 \cdot \alpha \cdot \sigma_{A_0}^2 - 2 \cdot (1-\alpha) \cdot \sigma_z^2 = 0$$

$$\Rightarrow \alpha = \frac{\sigma_z^2}{\sigma_{A_0}^2 + \sigma_z^2}$$

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE (continuare)

$$\alpha = \frac{\sigma_z^2}{\sigma_{A_0}^2 + \sigma_z^2}$$

- imaginea $A_0()$ nu este cunoscută, astfel încercăm să exprimăm pe α în funcție de $A()$, astfel:

$$\left.\begin{aligned} A &= A_0 + z \\ \sigma_A^2 &= \sigma_{A_0}^2 + \sigma_z^2 \end{aligned}\right\} \Rightarrow \alpha = \frac{\sigma_z^2}{\sigma_A^2}$$

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE (continuare)

$$\left.\begin{aligned} \alpha &= \frac{\sigma_z^2}{\sigma_A^2} = \frac{\sigma_z^2}{\sigma_{A_0}^2 + \sigma_z^2} \\ B &= \alpha \cdot \overline{A_0} + (1-\alpha)(A_0 + z) \end{aligned}\right\} \Rightarrow B_{optimal} = \frac{\sigma_z^2}{\sigma_A^2} \cdot \overline{A_0} + (1 - \frac{\sigma_z^2}{\sigma_A^2}) \cdot A$$

- cazuri limită:

$\sigma_z^2 = 0$ sau $\sigma_z^2 << \sigma_A^2 \Rightarrow \alpha \to 0 \Rightarrow B_{optimal} = A$

→ dacă zgomotul este redus sau ne aflăm în zone de contur, filtrul optimal este filtrul trece-tot.

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE (continuare)

- cazuri limită:

$$\alpha = \frac{\sigma_z^2}{\sigma_A^2} = \frac{\sigma_z^2}{\sigma_{A_0}^2 + \sigma_z^2},\ B_{optimal} = \frac{\sigma_z^2}{\sigma_A^2} \cdot \overline{A_0} + (1 - \frac{\sigma_z^2}{\sigma_A^2}) \cdot A$$

$\sigma_z^2 >> \sigma_{A_0}^2 \Rightarrow \alpha \to 1 \Rightarrow B_{optimal} = \overline{A}$

→ dacă zgomotul este important sau ne aflăm într-o regiune omogenă, filtrul optimal este filtrul trece-jos (de mediere).

Filtrarea adaptivă

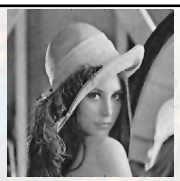
↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

A. adaptarea filtrelor liniare (continuare)

- filtrul LLMMSE, exemple:

imaginea inițială

imaginea cu zgomot gaussian (dispersie 10)



filtrul LLMMSE (5x5, dispersie=20)

filtrul LLMMSE (5x5, dispersie=10)

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

B. filtrul bilateral

idee: netezirea nu trebuie să aibă loc dacă pixelii nu sunt simiari.

bilateral= din două părți: ponderarea pixelilor se realizează de această dată în două moduri:

- *o pondere spațială*: ilustrează cât de aproape este un pixel de pixelul curent (central) → ponderi mai mari pixelilor apropiați,

- *o pondere a valorilor*: ilustrează cât de similari sunt pixelii vecini de pixelul curent (central) → ponderi mai mari pixelilor similari,

- ponderarea se face de regulă folosind funcții Gaussiene (distribuție normală),

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

B. filtrul bilateral (continuare)

- ponderile sunt alese astfel încât să păstreze condiția de normare pentru netezire.

- filtrarea se scrie:

$$B_{BL}(l,c) = \sum_{(m,n)\in V_{(l,c)}} w_{m,n}(l,c)\cdot A(l+m,c+n)$$

unde *A()* este imaginea inițială, *B()* este imaginea filtrată, $V_{(l,c)}$ reprezintă vecinătatea considerată pixelului curent *(l,c)* iar $w_{m,n}()$ reprezintă ponderile filtrului (valori reale),

$$w_{m,n}(l,c) = F(m,n,l,c)$$

Filtrarea adaptivă

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

B. filtrul bilateral (continuare)

$$B_{BL}(l,c) = \sum_{(m,n)\in V_{(l,c)}} w_{m,n}(l,c)\cdot A(l+m,c+n)$$

$$w_{m,n}(l,c) = \frac{w^{pos}_{m,n}(l,c)\cdot w^{val}_{m,n}(l,c)}{\sum_{(i,j)\in V_{(l,c)}} w^{pos}_{i,j}(l,c)\cdot w^{val}_{i,j}(l,c)}$$

$$w^{val}_{m,n}(l,c) = \frac{1}{\sigma_{val}\sqrt{2\pi}} e^{-\frac{(A(l+m,c+n)-A(l,c))^2}{2\sigma^2_{val}}},\quad w^{pos}_{m,n}(l,c) = \frac{1}{\sigma_{pos}\sqrt{2\pi}} e^{-\frac{m^2+n^2}{2\sigma^2_{pos}}}$$

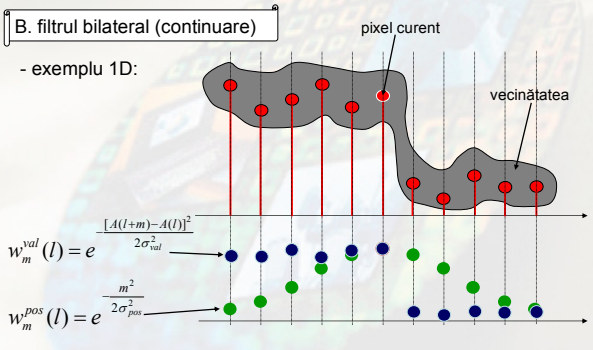
Filtrarea adaptivă

[M. Elad]

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

B. filtrul bilateral (continuare)

- exemplu 1D:



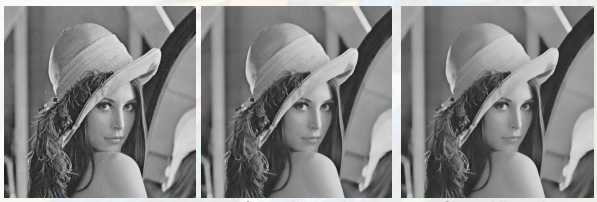
Filtrarea adaptivă

[M. Bertzen]

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

B. filtrul bilateral (continuare)

- exemple:



imaginea inițială | filtrare bilaterală σ_{pos}=3, σ_{val}=3 | filtrare bilaterală σ_{pos}=6, σ_{val}=3

Filtrarea adaptivă

[M. Bertzen]

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

B. filtrul bilateral (continuare)

- exemple (continuare):

imaginea inițială | filtrare bilaterală σ_{pos}=12, σ_{val}=3 | filtrare bilaterală, σ_{pos}=12, σ_{val}=6

Filtrarea adaptivă

[M. Bertzen]

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

B. filtrul bilateral (continuare)

- exemple (continuare):

imaginea inițială | filtrare bilaterală σ_{pos}=3, σ_{val}=3 | filtrare bilaterală, σ_{pos}=6, σ_{val}=3

Filtrarea adaptivă

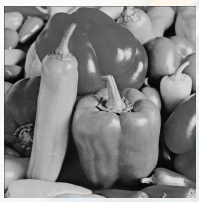
[M. Bertzen]

↳ **adaptarea coeficienților** funcțiilor de combinare a valorilor

B. filtrul bilateral (continuare)

facilități: - păstrează contururile,
- aplicabil pentru vecinătăți mari,
- poate fi iterat,

- exemple (continuare):



imaginea inițială

filtrare bilaterală
σ_{pos}=12, σ_{val}=3

filtrare bilaterală,
σ_{pos}=12, σ_{val}=6

Filtrarea adaptivă

↳ **adaptarea filtrelor intrinsec neliniare**

idee: determinarea automată a rangului statisticii de ordine în funcție de vecinătate:

$$B(l,c) = rank_j\{A_1, A_2, ..., A_K\},\ j = ?$$

→ Zamperoni, dacă vecinătatea în care este realizată filtrarea conține K pixeli, atunci rangul statisticii filtrului de ordine, *j*, poate fi estimat ca:

$$j = \left[\frac{1}{2} + \sum_{i=1}^{K} \frac{A_{(i)} - A_{(1)}}{A_{(K)} - A_{(1)}}\right] = \left[\frac{1}{2} + \sum_{i=1}^{K} \frac{A_{(i)} - \min}{\max - \min}\right]$$

Filtrarea adaptivă

↳ **adaptarea filtrelor intrinsec neliniare**

$$j = \left[\frac{1}{2} + \sum_{i=1}^{K} \frac{A_{(i)} - \min}{\max - \min}\right] = \left[\frac{1}{2} + \frac{\sum_{i=1}^{K} A_{(i)} - K \cdot \min}{\max - \min}\right]$$

- exemple:

imaginea inițială

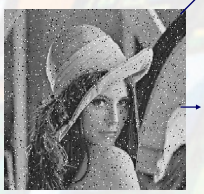
filtru Zamperoni (3x3)

Filtrarea adaptivă

↳ **adaptarea filtrelor intrinsec neliniare**

- exemple (continuare):

imaginea inițială



imaginea cu zgomot impulsiv (p=0.05)

filtru median (3x3)

filtru Zamperoni (3x3)

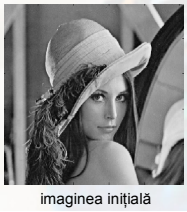
Filtrarea adaptivă

↳ **adaptarea filtrelor intrinsec neliniare**

- accentuare: extreme locale

$$B(l,c) = \begin{cases} A_{(K)} & A(l,c) \geq \dfrac{A_{(K)} + A_{(1)}}{2} \\ A_{(1)} & altfel \end{cases}$$

imaginea inițială



imaginea inițială

accentuare (3x3)

accentuare (3x3)

6.4. Evaluarea calității filtrării

Evaluarea calității filtrării

scopul principal al operațiilor de filtrare este dat de eliminarea perturbațiilor din imagine şi îmbunătățirea calității vizuale a acesteia (percepției).

→ cum ştim dacă un filtru este eficient sau nu ?

trebuie găsită o modalitate prin care să exprimăm cât de diferită este imaginea filtrată față de o imagine de *referință* (ex. imaginea perfectă neperturbată)

abordarea 1: evaluare vizuală

care dintre cele două imagini este mai bună ?

→ subiectivitate
persoane diferite pot avea percepții diferite

Evaluarea calității filtrării

abordarea 2: evaluare pe baza unei măsuri matematice

- dacă $A_0()$ reprezintă imaginea de referință (ex. neperturbată) şi $A()$ reprezintă imaginea filtrată sau perturbată, atunci putem calcula:

→ **eroarea pătratică medie** (Mean Squared Error – MSE):

$$MSE = \overline{(A - A_0)^2} = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[A(m,n) - A_0(m,n)\right]^2$$

→ **eroarea absolută medie** (Mean Absolute Error - MAE):

$$MAE = \overline{|A - A_0|} = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|A(m,n) - A_0(m,n)\right|$$

Evaluarea calității filtrării

abordarea 2: evaluare pe baza unei măsuri matematice (continuare)

→ **raportul semnal zgomot** (Signal to Noise Ratio - SNR):

$$SNR = 10 \cdot \log \frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left(A_0(m,n)\right)^2}{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[A(m,n) - A_0(m,n)\right]^2} [dB]$$

→ **raportul semnal de vârf zgomot** (Peak Signal to Noise Ratio - PSNR):

$$PSNR = 10 \cdot \log \frac{M \cdot N \cdot \max_{(m,n)}\{A_0^{\,2}(m,n)\}}{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[A(m,n) - A_0(m,n)\right]^2} [dB]$$

- evident, întotdeauna PSNR >= SNR

Evaluarea calității filtrării

abordarea 2: evaluare pe baza unei măsuri matematice (continuare)

cum trebuie să fie mărimile anterioare pentru a avea o calitate a imaginii cât mai bună ?

→ SNR, PSNR = cât mai mari

→ MSE, MAE = cât mai mici

SNR=20 dB

SNR=17 dB

SNR=14 dB

în general folosim SNR:
- calitate bună SNR > 25dB
- diferențe imperceptibile SNR > 30dB

Evaluarea calității filtrării

abordarea 2: evaluare pe baza unei măsuri matematice (continuare)

- exemple:

imagine inițială

imagine cu zgomot alb (disp. 13)

SNR = 17.8
PSNR = 23.1
MSE = 287
MAE = 13.5

filtru de mediere (3x3)

SNR = 21.5
PSNR = 26.8
MSE = 125
MAE = 7.6

Evaluarea calității filtrării

abordarea 2: evaluare pe baza unei măsuri matematice (continuare)

- exemple (continuare):

imagine inițială

imagine cu zgomot impulsiv (p=0.1)

SNR = 9.8
PSNR = 15.1
MSE = 1863
MAE = 12.6

filtru median (3x3)

SNR = 23.3
PSNR = 28.6
MSE = 84
MAE = 4.3

Evaluarea calității filtrării

abordarea 2: evaluare pe baza unei măsuri matematice (continuare)

- exemple (continuare):

imagine cu zgomot impulsiv (p=0.1)

SNR = 9.8
PSNR = 15.1
MSE = 1863
MAE = 12.6

filtru median (3x3)

SNR = 23.3
PSNR = 28.6
MSE = 84
MAE = 4.3

filtru de mediere (3x3)

SNR = 17.3
PSNR = 22.6
MSE = 330
MAE = 12.9

Evaluarea calității filtrării

abordarea 2: evaluare pe baza unei măsuri matematice (continuare)

în anumite situații, cu toate că eroarea matematică este mai mică la imaginea filtrată, din punct de vedere vizual, aceasta poate fi mai deteriorată (“deranjantă”) decât imaginea perturbată.

- care imagine este mai bună calitativ vizual ?

în practică, în funcție de aplicație se folosesc ambele tipuri de evaluari: subiective şi obiective.

Sfârşit Curs