Technical University of Cluj - Napoca
Computer Science Department

Procesarea Imaginilor

Curs 8:

Filtre digitale

Filtrarea digitala a imaginilor

Filtrarea digitală - una dintre cele mai utilizate tehnici în procesarea imaginilor ⇒ detecţie de muchii, eliminare zgomote, restaurare imagini

Două clase principale de filtre:

- **liniare**: ieşirea este o combinaţie liniară a valorilor pixelilor de intrare

  ⇒ se aplică prin operatorul de convoluție

  ex: suma, medie aritmetică, filtru Gaussian

- **neliniare**: ieşirea este o combinaţie neliniară a valorilor pixelilor de intrare

  ⇒ modul de aplicare este dependent de tipul filtrului

  ex: minim, maxim, element median

Technical University of Cluj Napoca
Computer Science Department

Filtre digitale liniare

Obiectiv: convoluţia imaginii *f(i,j)* cu funcţia de filtrare *h(i,j)*

În spaţiul Real (coordonate imagine):

$$g(i,j) = h(i,j) * f(i,j)$$

În spaţiul Fourier (spaţiul frecvenţelor) $\Rightarrow$ teorema convoluţiei:

$$G(k,l) = H(k,l)F(k,l)$$

Operaţia de filtrare este controlată de forma funcţiei de filtrare:

h(i,j) în spaţiul Real

H(k,l) în spaţiul Fourier

Cele două modalităţi de aplicare sunt identice matematic, dar diferă d.p.d.v. al efortului de calcul.

Technical University of Cluj Napoca
Computer Science Department

Convoluţia în spaţiul Fourier

$$g(i,j) = F^{-1}\{H(k,l)F(k,l)\}$$

Efortul de calcul:

2x DFT + 1x înmulţire complexă (operatii în virgulă mobilă)

- dacă *H(k,l)* se deduce din *h(i,j)* $\Rightarrow$ +1x DFT

Costul de calcul nu depinde de tipul (dimensiunea) filtrului *h(i,j)*

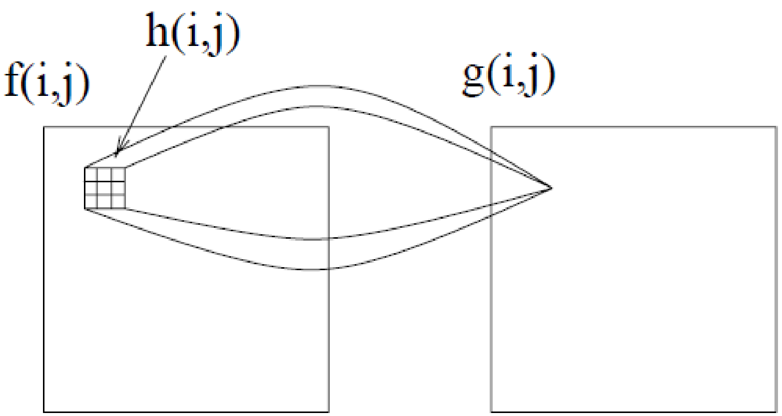
Se poate aplica doar pentru operaţii de filtrare liniare

NU se poate aplica pentru operaţii de filtrare ne-liniare

Convoluţia în spaţiul Real

Pentru un filtru *h(i,j)* de dimensiune MxM, convoluţia se defineşte ca:

$$g(i,j) = \sum_{m,n=-M/2}^{M/2} h(m,n) f(i-m, j-n)$$



Schema "shift & multiply"

Operaţii pe tip întreg / byte

Costul de calcul ~ M^2 (depinde de dimensiunea M a filtrului)

Technical University of Cluj Napoca
Computer Science Department

Convoluţie în spaţiul Real sau Fourier?

Costul convoluţiei în spaţiul Real depinde de dimensiunea filtrului

Costul convoluţiei în spatiul Fourier – fix (depinde de costul DFT şi de costul înmulţirii, nu de felul operaţiei)

În funcţie de tipul platformei hardware există o limită a dimensiunii filtrului sub care care aplicarea convoluţiei în spaţiul Real este mai eficientă decât aplicarea convoluţiei în spaţiul Fourier

Dacă se depăşeşte această limită pentru dimensiunea filtrului, aplicarea convoluţiei în spaţiul Fourier este mai eficientă (cost de calcul mai scăzut).

Technical University of Cluj Napoca
Computer Science Department

Filtrarea în spaţiul Fourier

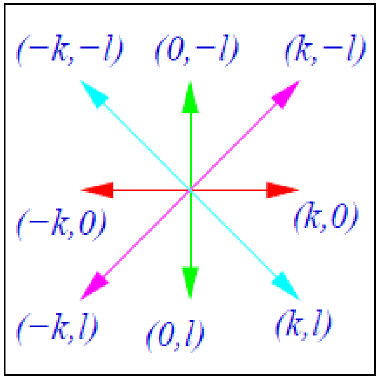
- Modifică transformata Fourier a imaginii de intrare: *F(k,l)*
- Depinde de forma filtrului: *H(k,l)*

Imaginile de intrare / ieşire ⇒ **Reale** (majoritatea aplicaţiilor)

Transformata Fourier complexă a imaginilor reale ⇒ proprietăţi de simetrie:

- Partea **Reala simetrică**

- Partea **Imaginară anti-simetrică**



Pentru ca imaginea de ieşire să fie Reală ⇒ filtrul Fourier trebuie să respecte proprietăţile de simetrie

Filtre trece jos (low-pass)

- Permit trecerea neatenuată a frecvenţelor spaţiale joase
- Atenuează sau blochează trecerea frecvenţelor înalte
- Folosite în reducerea zgomotului din imagini

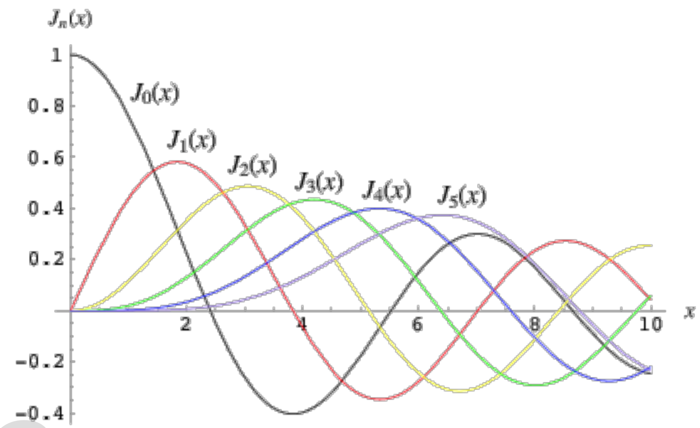
Filtrul trece jos ideal ⇒ blochează toate frecvenţele mai mari decat o frecvenţă de tăiere (cut-off frequency) w_0:

$$
\begin{aligned}
H(k,l) &= 1 \qquad k^2 + l^2 \leq w_0^2 \\
&= 0 \qquad \text{else}
\end{aligned}
$$

Filtrul trece jos în spaţiul Real:

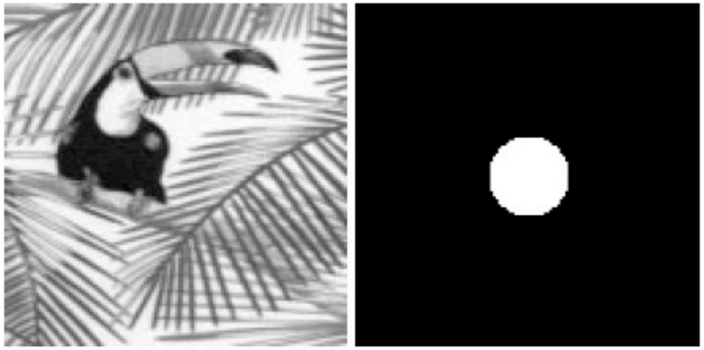
$$h(i,j) = \frac{J_1(r/w_0)}{r/w_0}$$

$$r^2 = i^2 + j^2.$$



Funcţii Bessel

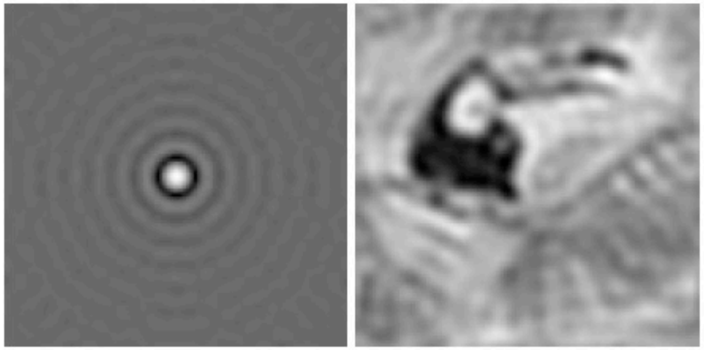
Filtrul trece jos ideal - exemplu



Input image Low-pass filter

Dimensiune imagine: 128 x 128

Frecvenţă tăiere: 15 pixeli



Real space filter Filtered Image

Imaginea rezultată după aplicarea filtrului trece-jos ideal are aspect de undă circulară la variaţii bruşte ale intensităţii din imagine ⇒ nefolositor

Filtrul trece jos Gaussian

$$H(k,l) = \exp\left(-\left(\frac{w}{w_0}\right)^2\right) \qquad w^2 = k^2 + l^2$$

Lăţimea filtrului w_0 ($k^2 + l^2 = w_0^2$) este valoarea pentru care Gaussianul H(k,l) este 1/e.

În spaţiul real se obţine tot un filtru Gaussian:

$$h(i,j) = \frac{\pi}{w_0^2} \exp\left(-\pi^2 w_0^2 r^2\right) \qquad r^2 = i^2 + j^2.$$

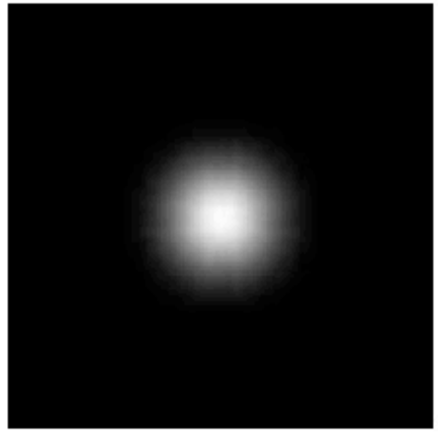
Filtrul este infinit atât în spaţiul Fourier cât şi în cel Real $\Rightarrow$ atenuează frecvenţele înalte, fără să le înlăture complet ca şi filtrul ideal.

Efecte: netezirea imaginii ("smoothing"), fără efecte de unda circulară ("ringing").

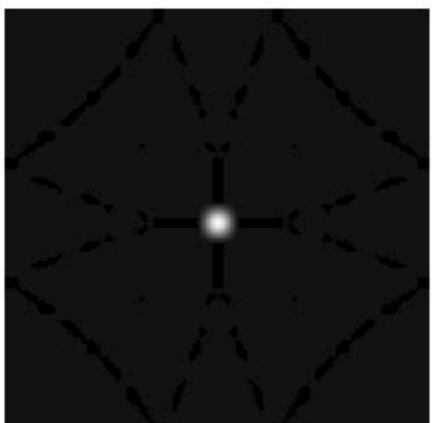
De obicei se aplică înainte de o segmentare (bazată pe regiuni sau pe muchii) pentru eliminarea zgomotului (frecvenţe înalte)

Filtrul trece jos Gaussian - exemplu

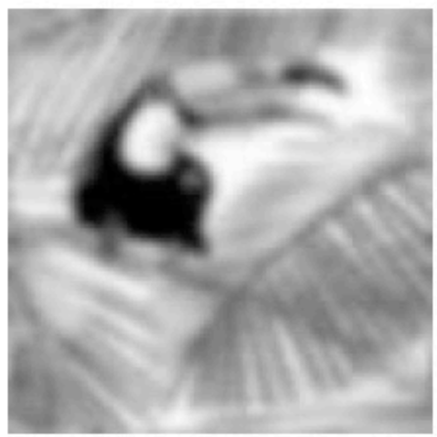
(a) Input image



(b) Low-pass filter



(c) Filter in real space.



(d) Filtered Image

Dimensiune imagine:
128 x 128

w_0 = 30 pixeli

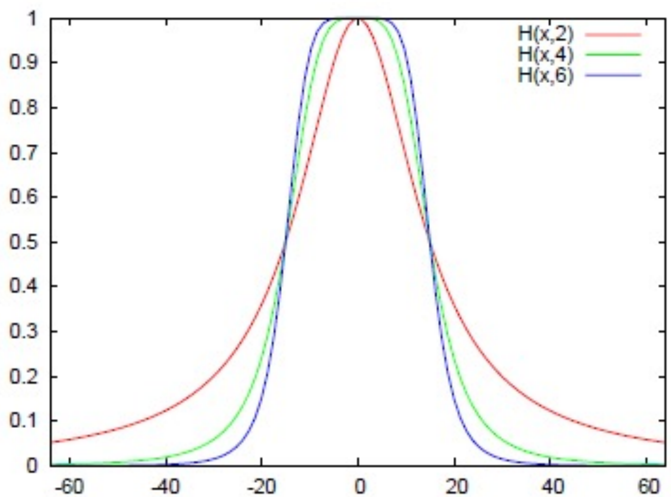
Alte fitre trece jos

Filtrul Butterworth

$$H(k,l) = \frac{1}{1 + (\frac{w}{w_0})^n}$$

w_0 - punctul în care H(k,l) = ½

n – ordinul filtrului



Filtrul trapezoidal

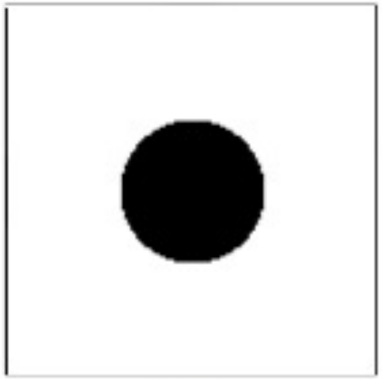
$$\begin{aligned} H(k,l) &= 1 && \text{for } w < w_0 \\ &= \frac{(w-w_1)}{(w_0-w_1)} && \text{for } w_0 \le w \le w_1 \\ &= 0 && \text{for } w > w_1 \end{aligned}$$

Introduce efect de undă circulară (mai puternic decât cel Gaussian sau Butterworth, dar mai slab decât cel Ideal)

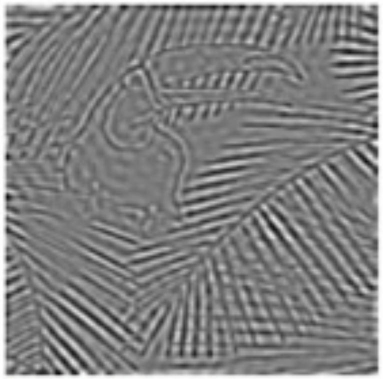
Filtre trece sus (high-pass)

- Permit trecerea neatenuată a frecvenţelor spaţiale înalte
- Atenuează sau blochează trecerea frecvenţelor joase
- Folosit la accentuarea zonelor de tranziţie (muchii)

Filtrul trece sus ideal $\Rightarrow$ blochează toate frecvenţele mai mici decât o frecvenţă de tăiere (cut-off frequency) w_0:



Filter



Output Image

$$H(k,l) = \begin{cases} 1, & daca\ k^2 + l^2 > {w_0}^2 \\ 0, & in\ caz\ contrar \end{cases}$$

Imaginea rezultată după aplicarea filtrului trece-sus ideal are aspect de undă circulară $\Rightarrow$ nefolositor

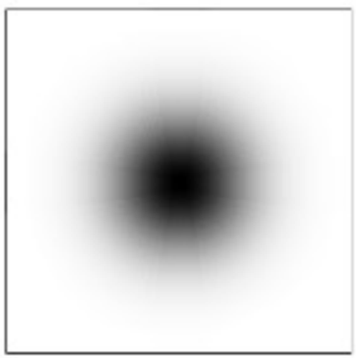
$w_0 = 25$

Filtrul trece sus Gaussian

- Realizează o reducere graduală a frecvenţelor joase
- Frecvenţele înalte trec prin filtru nealterate

$$H(k,l) = 1 - \exp\left(-\left(\frac{w}{w_0}\right)^2\right) \qquad w^2 = k^2 + l^2,$$

În spaţiul Real $\Rightarrow$ filtru *h(i,j)* cu variaţie graduală $\Rightarrow$ accentuarea frecvenţelor înalte (ex: muchii) fără efect de unda circulară ("ringing")



Filter

Output Image

Acest filtru se poate combina cu un Gaussian trece jos $\Rightarrow$ Difference of Gaussian (DOG)

$w_0 = 25$

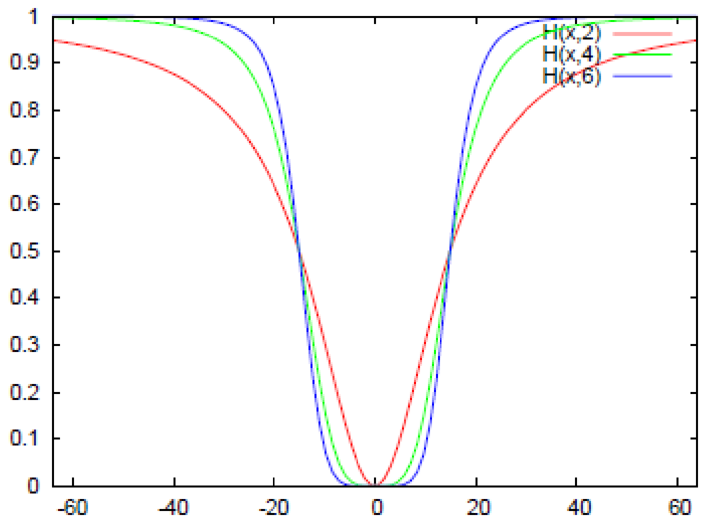
Alte filtre trece sus

Filtrul Butterworth trece sus

$$H(k,l) = 1 - \frac{1}{1 + \left(\frac{w}{w_0}\right)^n} = \frac{1}{1 + \left(\frac{w_0}{w}\right)^n}$$

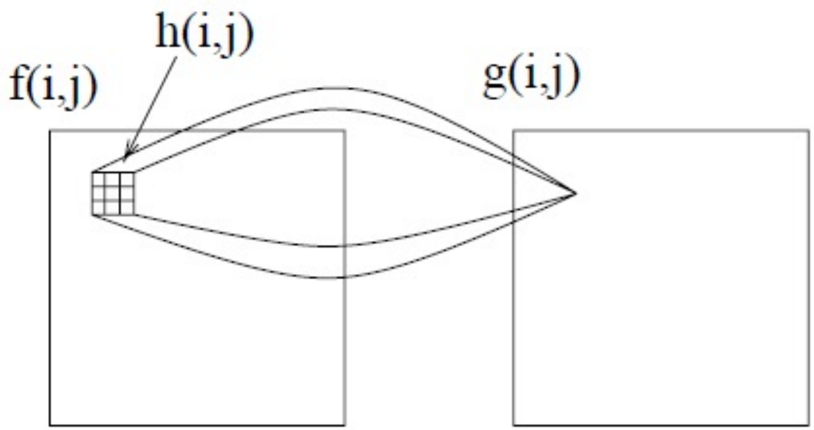
w_0 - punctul în care H(k,l) = ½

n – ordinul filtrului



Filtrarea în spaţiul Real

Filtrul este specificat in spatiul Real printr-un nucleu h(i,j) de dimensiune finită: 3x3, 5x5, 7x7 ….



Elementele h(i,j) sunt reale (întregi sau virgulă mobilă)

Pentru nuclee mai mari de 7x7 folosirea filtrării în spaţiul Fourier poate fi mai rapidă

Filtre de medie în spaţiul Real

Înlocuiesc fiecare pixel cu media vecinilor ⇒ efect low-pass (reducere a zgomotului)

Exemple:

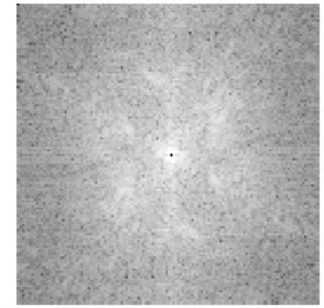
5 Pixel Average

$$\begin{matrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{matrix}$$

9 Pixel Average

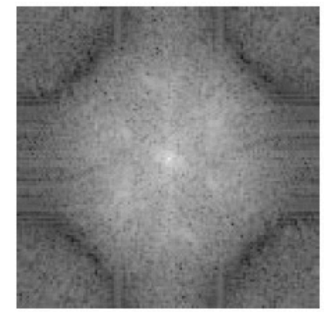
$$\begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Input Image



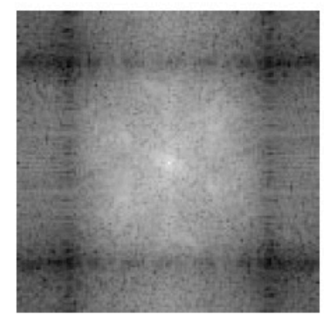
Fourier Transform

5 point ave



Fourier Transform

9 point ave



Fourier Transform

Derivata în spaţiul real

Derivata unei funcţii continue 1D:

$$\frac{df(x)}{dx} = \lim_{\delta \to 0} \frac{f(x+\delta) - f(x)}{\delta}$$

Pentru cazul discret ($\delta = 1$): $$\frac{df(i)}{di} = f(i+1) - f(i)$$

Procesul de derivare se poate exprima printr-o operaţie de convoluţie:

$$\frac{df(i)}{di} = [-1 \quad 1] \odot f(i)$$

Pentru ($\delta = 2$):

$$\frac{df(i)}{di} = [-1 \quad 0 \quad 1] \odot f(i)$$

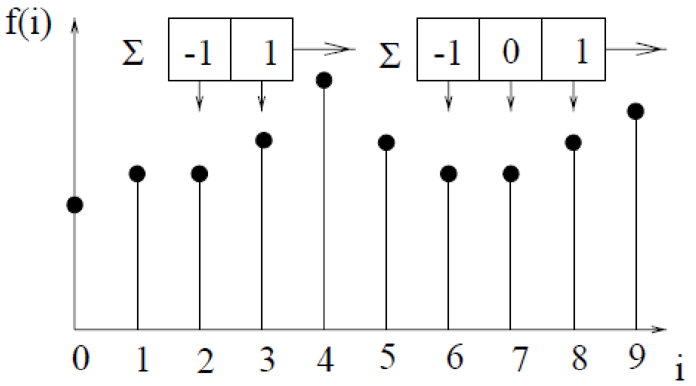


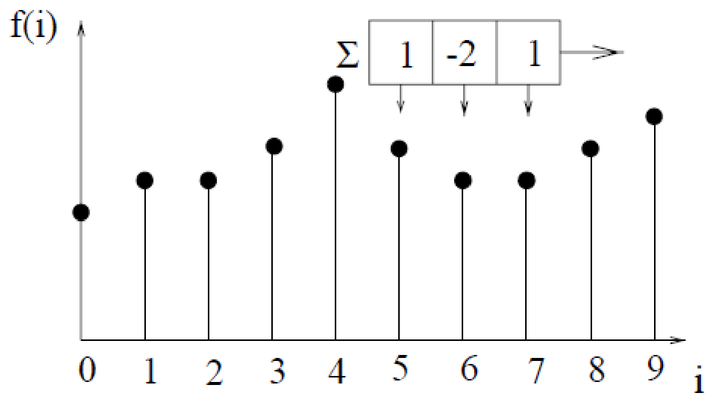
IMAGE PROCESSING

Derivata de ordin 2

Derivata de ordin 2 a unei funcţii discrete 1D:

$$\frac{d^2 f(i)}{di^2} = f(i+1) - 2f(i) + f(i-1)$$

Exprimată prin convoluţie: $$\frac{d^2 f(i)}{di^2} = [1 \quad -2 \quad 1] \odot f(i)$$



Datorită faptului că operaţia de convoluţie este asociativă, remarcăm că:

$$[1 \quad -2 \quad 1] = [-1 \quad 1] \odot [-1 \quad 1]$$

Technical University of Cluj Napoca
Computer Science Department

Derivate 2D

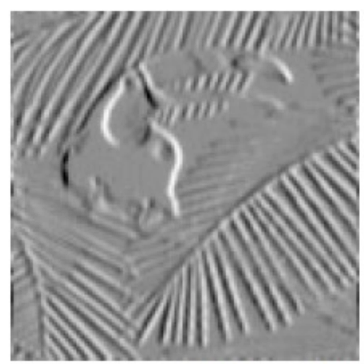
În cazul bi-dimensional (imagini) avem:

$$\frac{\partial f(i,j)}{\partial i} = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \odot f(i,j) \qquad \frac{\partial f(i,j)}{\partial j} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \odot f(i,j)$$

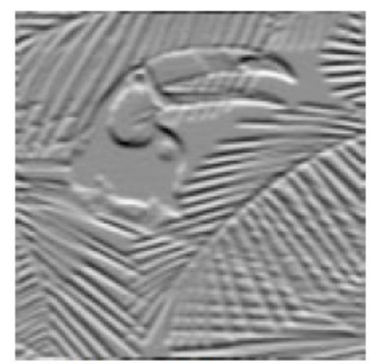
Pentru atenuarea zgomotului se practică medierea derivatelor pe 3 linii / coloane:

$$\frac{\partial f(i,j)}{\partial i} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \odot f(i,j) \qquad \frac{\partial f(i,j)}{\partial j} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \odot f(i,j)$$

Efecte: evidentierea muchiilor verticale / orizontale

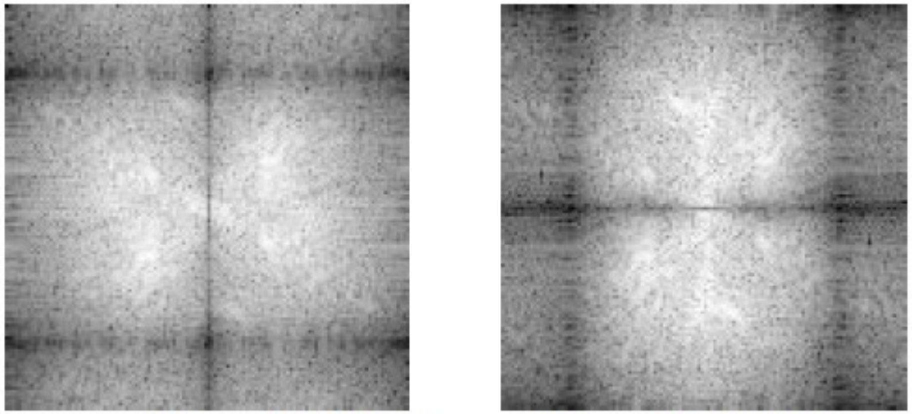


x-differential



y-differential

Derivate 2D



x-differential (FT) y-differential (FT)

Pe lângă liniile “negre” (zero) orizontale şi verticale centrale mai apar linii de “zero” secundare datorită efectului de mediere).

Derivata în spaţiul Fourier

Proprietăţile transformatei Fourier:

$$F\left\{\frac{\partial f(x,y)}{\partial x}\right\} = i2\pi u F(u,v) \qquad F\left\{\frac{\partial f(x,y)}{\partial y}\right\} = i2\pi v F(u,v)$$

$\Rightarrow$ Derivarea în spaţiul Fourier $\equiv$ înmultire cu *i2πu* sau *i2πv*

$\Rightarrow$ Efect high-pass (accentuarea frecvenţelor înalte)

Technical University of Cluj Napoca
Computer Science Department

Derivata de ordin 2

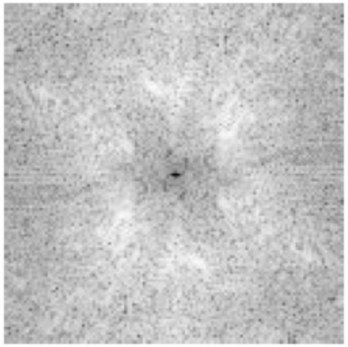
$$\frac{\partial^2 f(i,j)}{\partial i^2} = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} \odot f(i,j) \qquad \frac{\partial^2 f(i,j)}{\partial j^2} = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \odot f(i,j)$$

Laplacianul: $\nabla^2 f(i,j) = \frac{\partial^2 f(i,j)}{\partial i^2} + \frac{\partial^2 f(i,j)}{\partial j^2} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \odot f(i,j)$

Proprietăţile transformatei Fourier:

$$F\left\{\nabla^2 f(x,y)\right\} = -(2\pi w)^2 F(u,v) \qquad w^2 = u^2 + v^2,$$

Efecte ⇒ accentuarea muchiilor din imagine în toate direcţiile



Technical University of Cluj Napoca
Computer Science Department

Variaţii ale Laplacianului

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$ $\Rightarrow$ Laplacian mai puţin sensibil la zgomot

Accentuarea muchiilor: scăderea din imagine a Laplacianului

$$f(i,j) - \nabla^2 f(i,j) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \odot f(i,j)$$

Input image

Edge Enhanced

Utilizarea filtrelor liniare

Filtre low-pass: netezirea imaginilor, diminuarea efectelor zgomotelor (se folosesc de obicei înaintea aplicării detectorilor de puncte de muchie)

Filtre high-pass (filtre derivative): accentuarea frecvenţelor înalte (muchii).

Filtrele pot fi combinate (ex: formarea de filtre trece bandă - band-pass).

Datorită caracterului liniar, combinarea se poate face prin înmultire în spaţiul Fourier, şi prin convoluţie în spaţiul Real.

Filtre neliniare în spaţiul Real

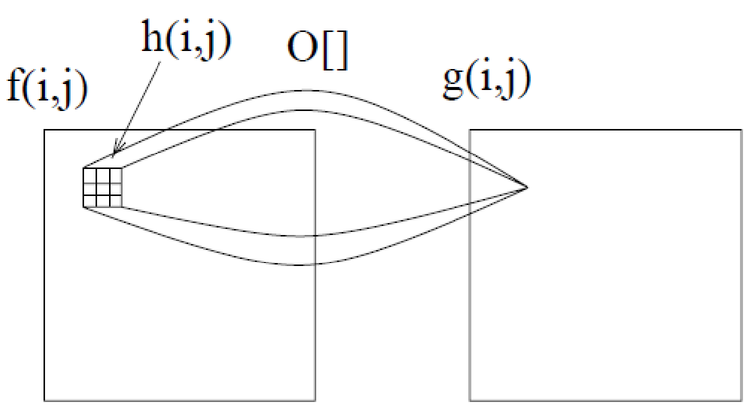
$$g(i,j) = O_{m,n \in w}[h(m,n) f(i-m, j-n)]$$

Domeniul de definiţie al măştii *h(m,n)* este dat de *w.*

Operaţia de filtrare este definită de masca *h(i,j)* şi de operatorul *O[]*.

La majoritatea filtrelor neliniare avem: $h(i,j) = 1 \quad i,j \in w$,

operaţia de filtrare fiind controlată doar de operatorul *O[]*



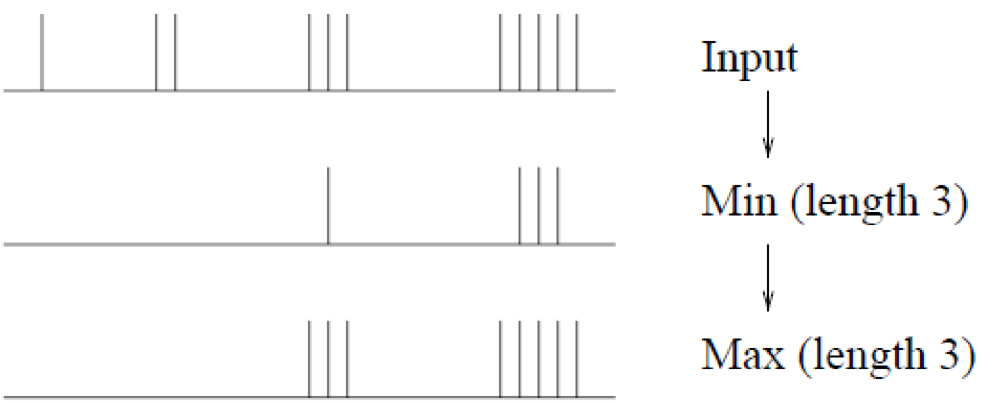
Filtre “shrink & expand”

Shrink = îngustare; Expand = lărgire

O[] = Min[] ⇒ shrink: obiectele luminoase (alb = obiect) sunt reduse în dimensiune cu un ordin de mărime egal cu dimensiunea filtrului

O[] = Max[] ⇒ expand: obiectele luminoase sunt extinse în dimensiune cu un ordin de mărime egal cu dimensiunea filtrului

⇒ Cele doua filtre sunt folosite in pereche pe imagini binare pentru eliminarea regiunilor mici / izolate



Obs: aplicarea filtrelor nu este comutativă:

$$E[S[f(i,j)]] \neq S[E[f(i,j)]]$$

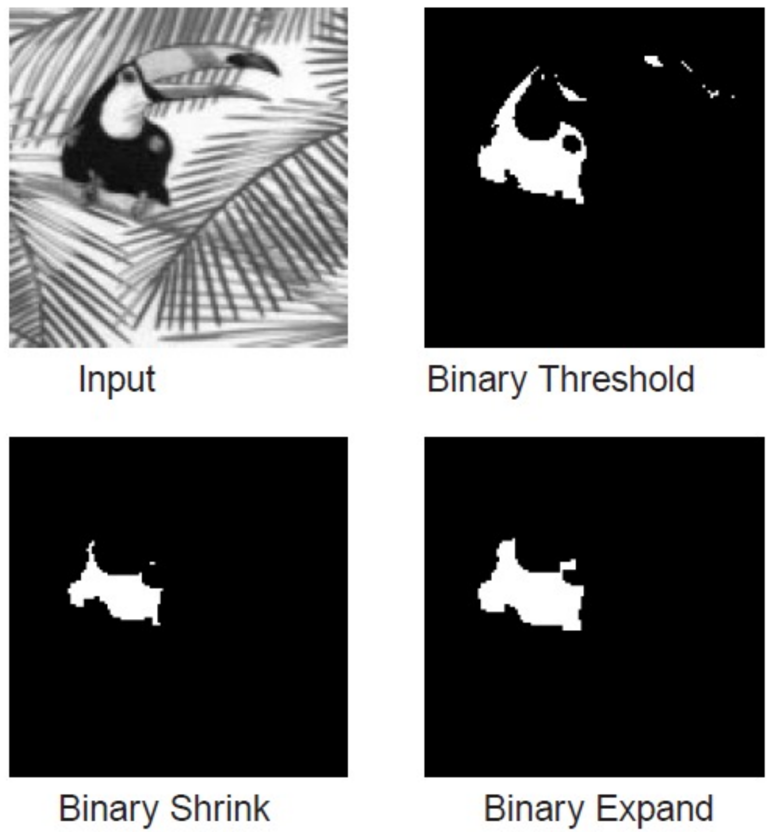
Technical University of Cluj Napoca
Computer Science Department

Filtre shrink/expand 2D

Exemplu: fereastră 2D de dimensiune 3x3

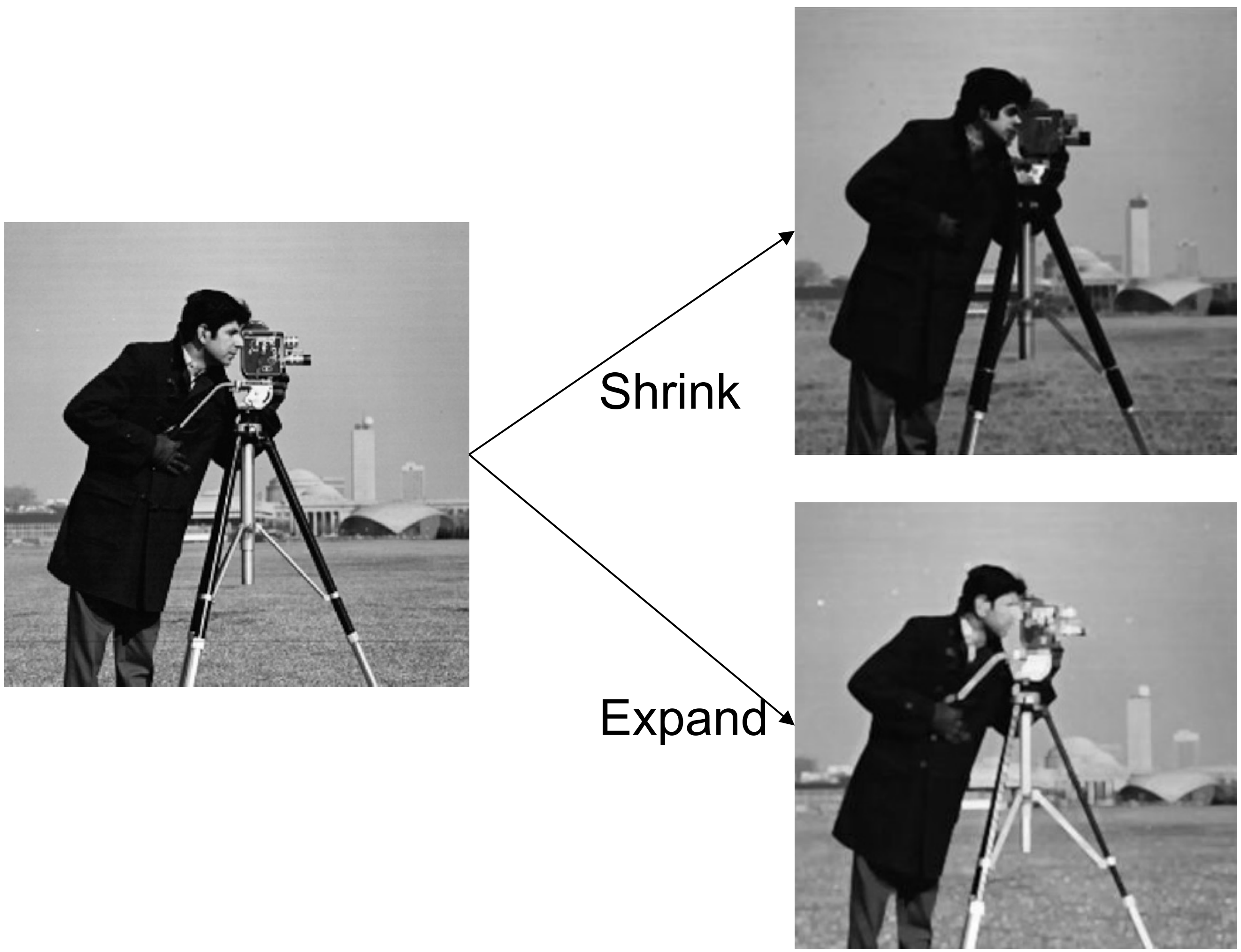
Efect: eliminarea obiectelor (obiect = pixel alb) mici/izolate dintr-o imagine binară

Exemplu filtrare shrink&expand cu un filtru de dimensiune 3x3:



Filtre shrink/expand 2D

Aplicarea pe imagini grayscale, fără binarizare:



Filtrul Threshold Average

Compară fiecare pixel cu media vecinilor (exclusiv valoarea lui) şi netezeşte (înlocuieşte pixelul curent) doar dacă diferenţa este semnificativă.

Pentru fiecare pixel se calculează media vecinilor:

$$A = \sum_{m,n=-M/2}^{M/2} h(m,n) f(i-m, j-n)$$

Ex. - pt. un filtru 3x3: $h(i,j) = \begin{bmatrix} k & k & k \\ k & 0 & k \\ k & k & k \end{bmatrix}$ $\quad k = 1/(M^2 - 1) = 0.125$

$$g(i,j) \begin{array}{ll} = A & |A - f(i,j)| > T \\ f(i,j) & \text{else} \end{array}$$

Stabilirea T: “trial and error” (~ fracţiune din (Max(f) - Min(f))

Filtrul median

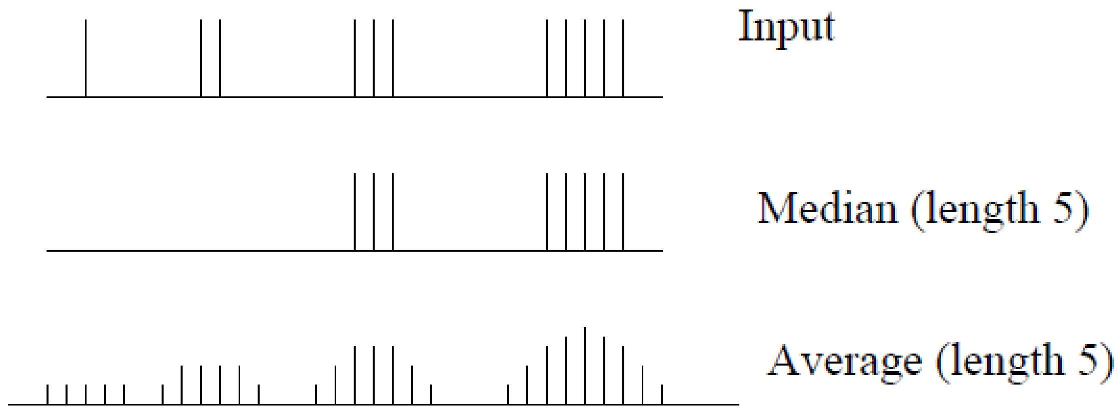
$$O[] = \text{Median}[]$$

Median := elementul de mijloc dintr-o mulţime sortată

Exemplu:

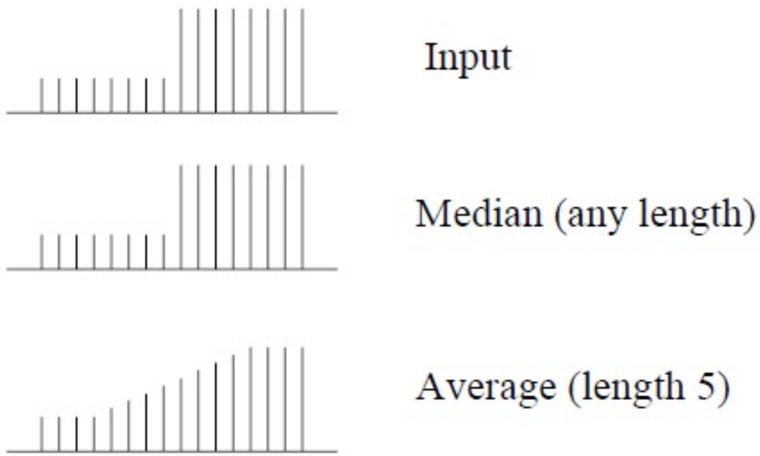
$$f(i) = 61, 10, 9, 11, 9 \qquad \text{Median}[f(i)] = 10$$

În cazul 1D filtrul median elimină toate trăsăturile/obiectele cu dimensiune mai mică decât M/2+1, păstrând toate celelalte trăsături



Proprietăţi (filtrul median)

Netezire fără alterarea punctelor de muchie (low-pass selectiv):



În cazul 2D elimină toate trăsăturile (obiectele) cu dimensiunea ("aria") mai mică de $M^2/2-1$, păstrându-le pe toate celelalte.

3 × 3 Median

5 × 5 Median