



**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 ⇒ detecție de muchii, eliminare zgomote, restaurare imagini

Două clase principale de filtre:

- **liniare**: ieșirea este o combinație liniară a valorilor pixelilor de intrare

⇒ se aplică prin operatorul de convoluție

ex: suma, medie aritmetică, ...

- **neliniare**: ieșirea este o combinație neliniară a valorilor pixelilor de intrare

⇒ modul de aplicare este dependent de tipul filtrului

ex: minim, maxim, element median



# Filtre digitale liniare

---

Obiectiv: convoluția imaginii  $f(i,j)$  cu funcția de filtrare  $h(i,j)$

În spațiul Real (coordonate imagine):

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

În spațiul Fourier (spațiul frecvențelor)  $\Rightarrow$  teorema convoluției:

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

Operația de filtrare este controlată de forma funcției de filtrare:

$h(i,j)$  în spațiul Real

$H(k,l)$  în spațiul Fourier

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



# Convoluția în spațiul Fourier

---

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

Efortul de calcul:

2x DFT + 1x înmulțire 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 operații de filtrare liniare

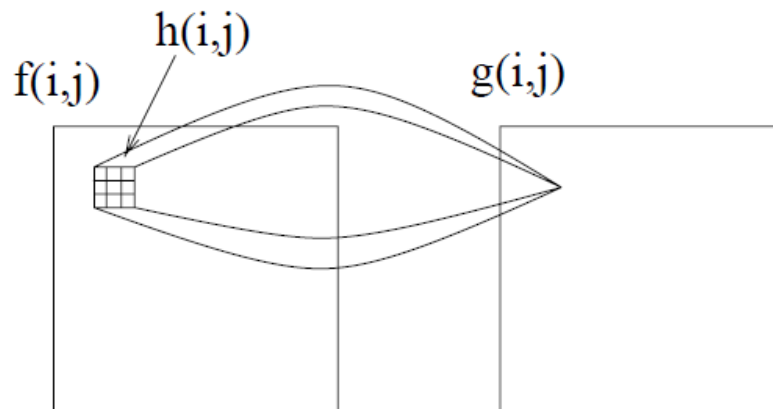
NU se poate aplica pentru operații de filtrare ne-liniare



# Convoluția în spațiul Real

Pentru un filtru  $h(i,j)$  de dimensiune  $M \times M$ , convoluția se definește ca:

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



Schema “shift & multiply”

Operații pe tip întreg / byte

Costul de calcul  $\sim M^2$  (depinde de dimensiunea  $M$  a filtrului)



# Convoluție în spațiul Real sau Fourier?

---

Costul convoluției în spațiul Real depinde de dimensiunea filtrului

Costul convoluției în spațiul Fourier – fix (depinde de costul DFT și de costul înmulțirii, nu de felul operației)

În funcție de tipul platformei hardware există o limită a dimensiunii filtrului sub care aplicarea convoluției în spațiul Real este mai eficientă decât aplicarea convoluției în spațiul Fourier

Dacă se depășește această limită pentru dimensiunea filtrului, aplicarea convoluției în spațiul Fourier este mai eficientă (cost de calcul mai scăzut).



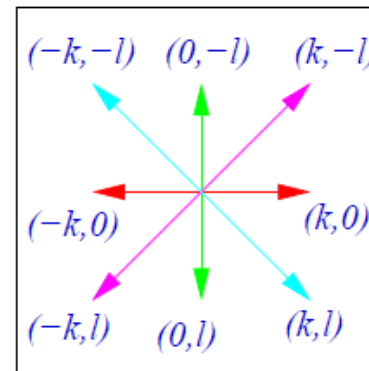
# Filtrarea în spațiul Fourier

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

Imaginile de intrare / ieșire  $\Rightarrow$  **Reale** (majoritatea aplicațiilor)

Transformata Fourier complexă a imaginilor reale  $\Rightarrow$  **proprietăți de simetrie:**

- Partea **Reala simetrică**
- Partea **Imaginară anti-simetrică**



**Pentru ca imaginea de ieșire să fie Reală  $\Rightarrow$  filtrul Fourier trebuie să respecte proprietățile de simetrie**



# Filtre trece jos (low-pass)

- Permit trecerea neatenuată a frecvențelor spațiale joase
- Atenuază sau blochează trecerea frecvențelor înalte
- Folosite în reducerea zgomotului din imagini

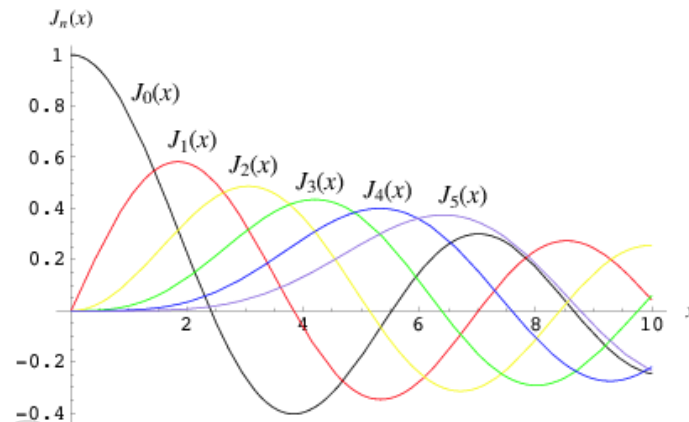
**Filtrul trece jos ideal**  $\Rightarrow$  blochează toate frecvențele mai mari decât o frecvență de tăiere (cut-off frequency)  $\omega_0$ :

$$H(k, l) = \begin{cases} 1 & k^2 + l^2 \leq \omega_0^2 \\ 0 & \text{else} \end{cases}$$

Filtrul trece jos în spațiul Real:

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

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

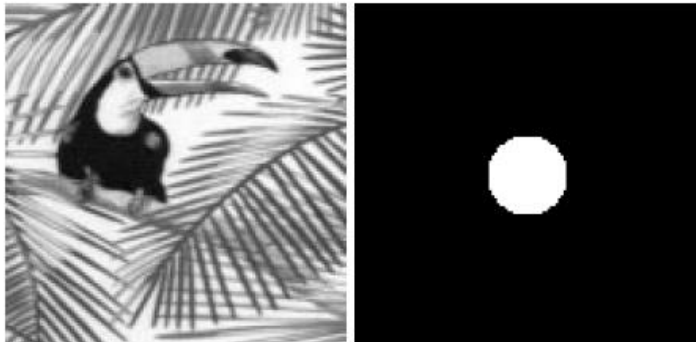


Funcții Bessel





## Filtrul trece jos ideal - exemplu

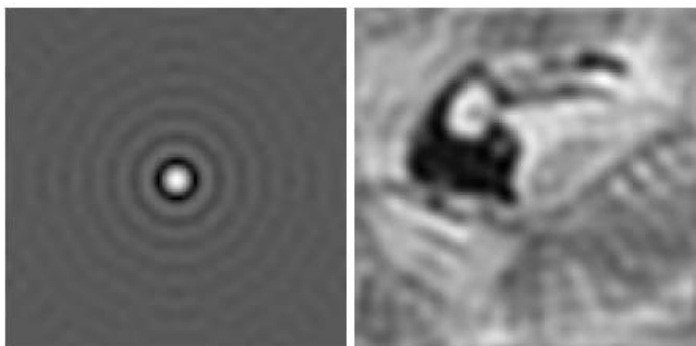


Input image

Low-pass filter

Dimensiune imagine: 128 x 128

Frecvență tăiere: 15 pixeli



Real space filter

Filtered Image

Imaginea rezultată după aplicarea filtrului trece-jos ideal are aspect de undă circulară la variații bruște ale intensității din imagine  $\Rightarrow$  nefolositor



# Filtrul trece jos Gaussian

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

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

În spațiul Real se obține tot un filtru Gaussian:

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

Filtrul este infinit atât în spațiul Fourier cât și în cel Real  $\Rightarrow$  atenuează frecvențele înalte, fără să le înlăture complet ca și 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 (frecvențe înalte)



# Filtrul trece jos Gaussian - exemplu



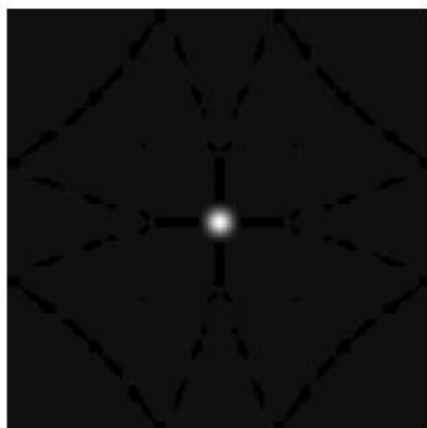
(a) Input image



(b) Low-pass filter

Dimensiune imagine:  
128 x 128

$w_0 = 30$  pixeli



(c) Filter in real space.



(d) Filtered Image

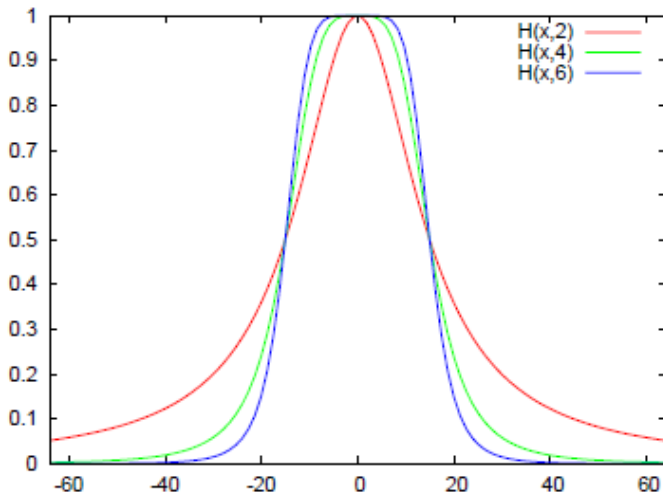


## Filtrul Butterworth

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

$w_0$  - punctul în care  $H(k,l) = \frac{1}{2}$

$n$  – ordinul filtrului



## Filtrul trapezoidal

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

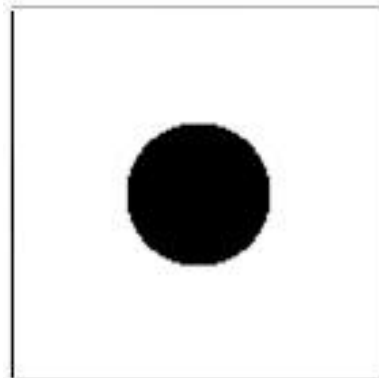
Introducece efect de undă circulară (mai puternic decât cel Gaussian sau Butterworth, dar mai slab decât cel Ideal)



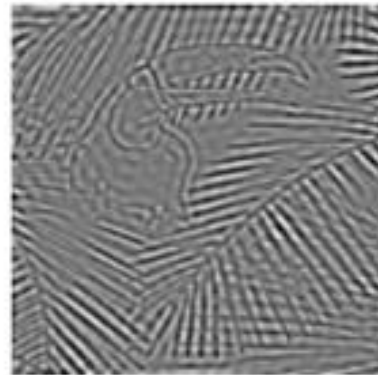
## Filtre trece sus (high-pass)

- Permit trecerea neatenuată a frecvențelor spațiale înalte
- Atenuază sau blochează trecerea frecvențelor joase
- Folosit la accentuarea zonelor de tranziție (muchii)

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



Filter



Output Image

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

$$w_0 = 25$$

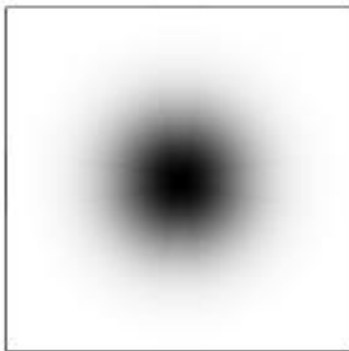


# Filtrul trece sus Gaussian

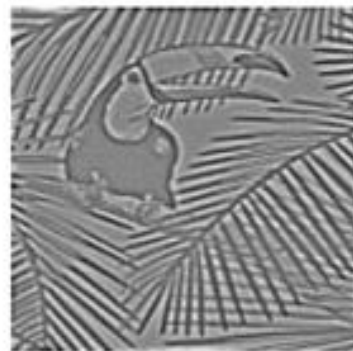
- Realizează o reducere graduală a frecvențelor joase
- Frecvențele înalte trec prin filtru nealterate

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

În spațiul Real  $\Rightarrow$  filtru  $h(i, j)$  cu variație graduală  $\Rightarrow$  accentuarea frecvențelor 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)

$$w_0 = 25$$



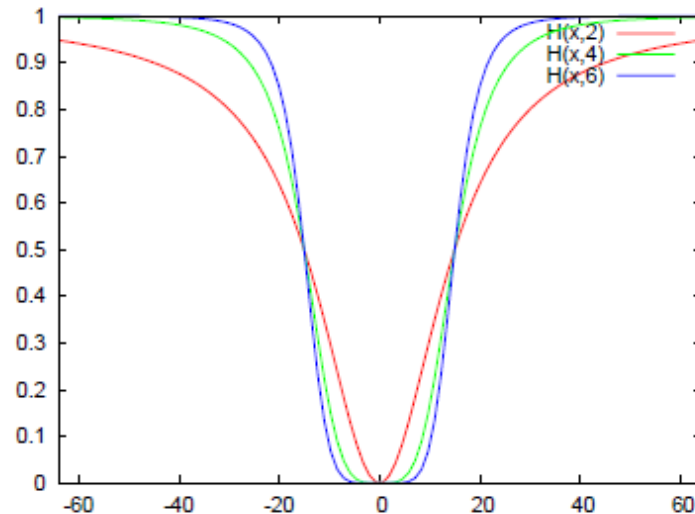
# Alte filtre trece sus

## 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  $H(k, l) = \frac{1}{2}$

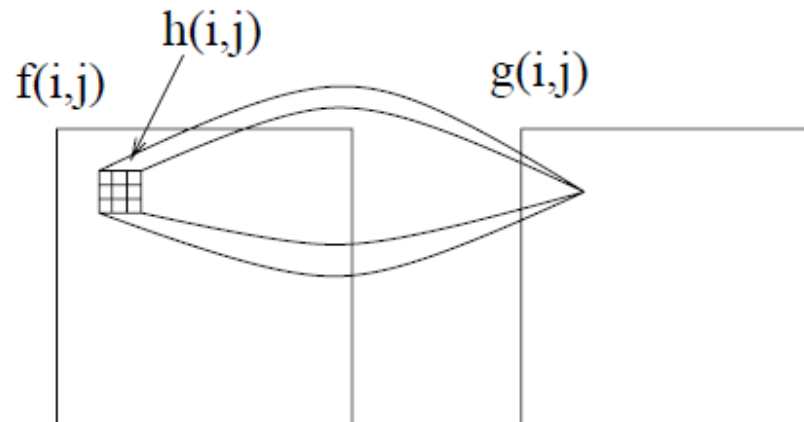
$n$  – ordinul filtrului





# Filtrarea în spațiul Real

Filtrul este specificat în spațiul Real printr-un nucleu  $h(i,j)$  de dimensiune finită:  $3 \times 3$ ,  $5 \times 5$ ,  $7 \times 7$  ....



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

Pentru nuclee mai mari de  $7 \times 7$  folosirea filtrării în spațiul Fourier poate fi mai rapidă





# Filtre de medie în spațiul Real

Înlocuiesc fiecare pixel cu media vecinilor  $\Rightarrow$  efect low-pass (reducere a zgomotului)

Exemple:

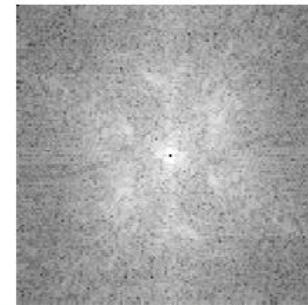
**5 Pixel Average**

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

**9 Pixel Average**

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

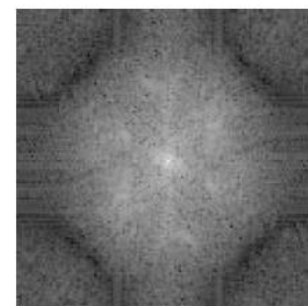

Input Image



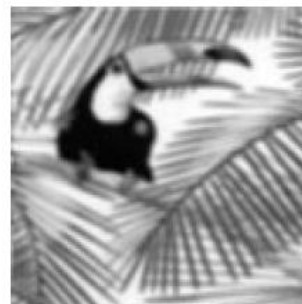
Fourier Transform



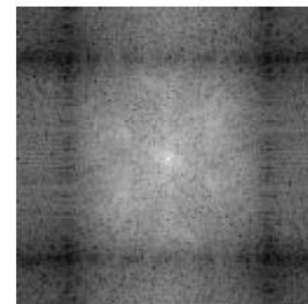
5 point ave



Fourier Transform



9 point ave



Fourier Transform



# Derivata în spațiul real

Derivata unei funcții continue 1D:

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

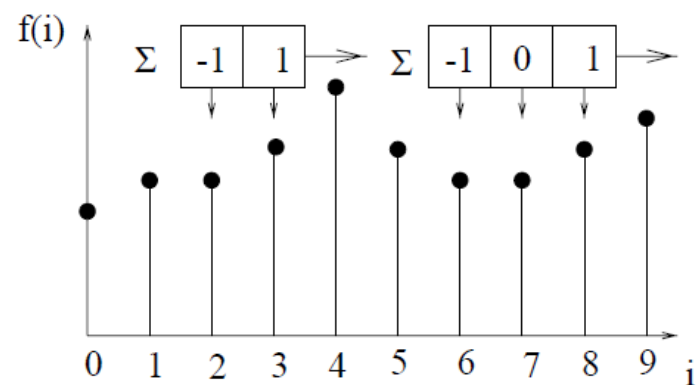
Pentru cazul discret ( $\delta = 1$ ):  $\frac{df(i)}{di} = f(i+1) - f(i)$

Procesul de derivare se poate exprima printr-o operație de convoluție:

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

Pentru ( $\delta = 2$ ):

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



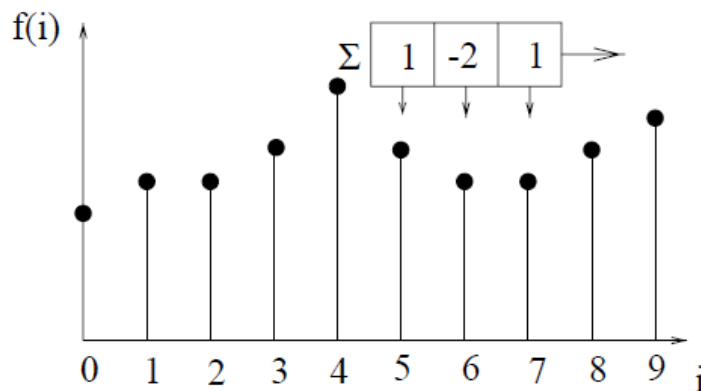


## Derivata de ordin 2

Derivata de ordin 2 a unei funcții discrete 1D:

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

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



Datorită faptului că operația de convoluție este liniară, remarcăm că:

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



## Derivate 2D

În cazul bi-dimensional (imagini) avem:

$$\frac{\partial f(i, j)}{\partial i} = [-1 \quad 0 \quad 1] \odot f(i, j)$$

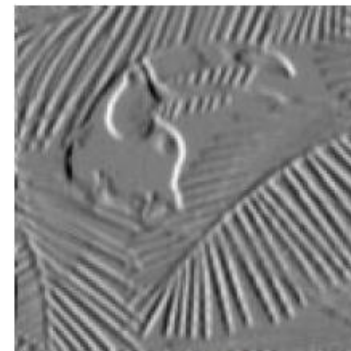
$$\frac{\partial f(i, j)}{\partial j} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \odot f(i, j)$$

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

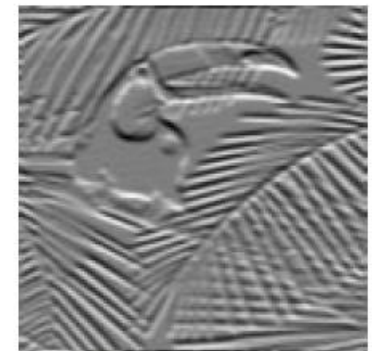
$$\frac{\partial f(i, j)}{\partial i} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \odot f(i, j)$$

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

Efecte: evidențierea muchiilor  
verticale / orizontale



x-differential



y-differential



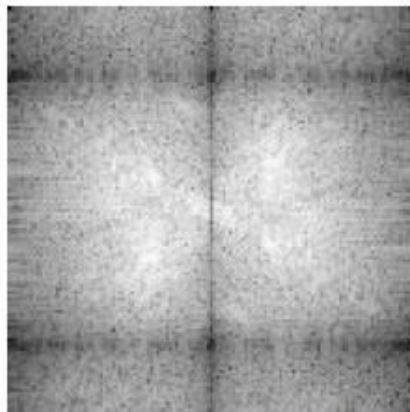
# Derivata în spațiul Fourier

Proprietățile transformatei Fourier:

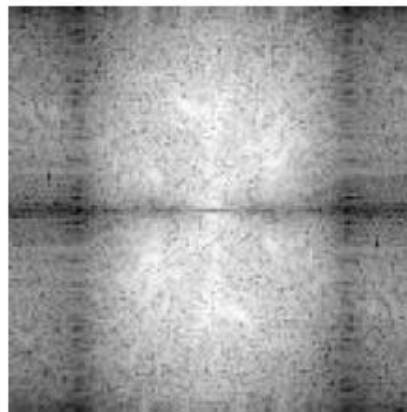
$$F \left\{ \frac{\partial f(x,y)}{\partial x} \right\} = i2\pi u F(u,v) \qquad F \left\{ \frac{\partial f(x,y)}{\partial y} \right\} = i2\pi v F(u,v)$$

⇒ Derivarea în spațiul Fourier  $\equiv$  înmulțire cu  $i2\pi u$  sau  $i2\pi v$

⇒ Efect high-pass (accentuarea frecvențelor înalte)



x-differential (FT)



y-differential (FT)

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



## Derivata de ordin 2

$$\frac{\partial^2 f(i, j)}{\partial i^2} = [1 \quad -2 \quad 1] \odot f(i, j)$$

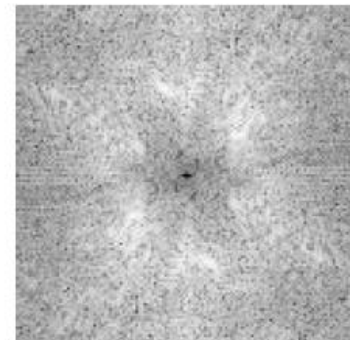
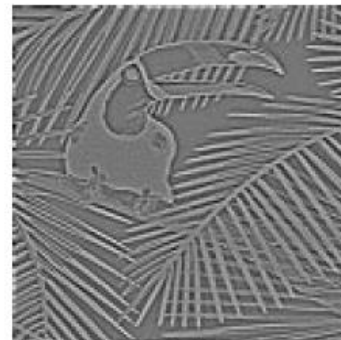
$$\frac{\partial^2 f(i, j)}{\partial j^2} = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \odot f(i, j)$$

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

Proprietățile transformatei Fourier:

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

Efecte  $\Rightarrow$  accentuarea  
muchiilor din imagine în toate  
direcțiile





# Variații ale Laplacianului

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

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 înainte de aplicarea detectorilor de puncte de muchie)

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

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

Datorită caracterului liniar, combinarea se poate face prin înmulțire în spațiul Fourier, și prin convoluție în spațiul Real.





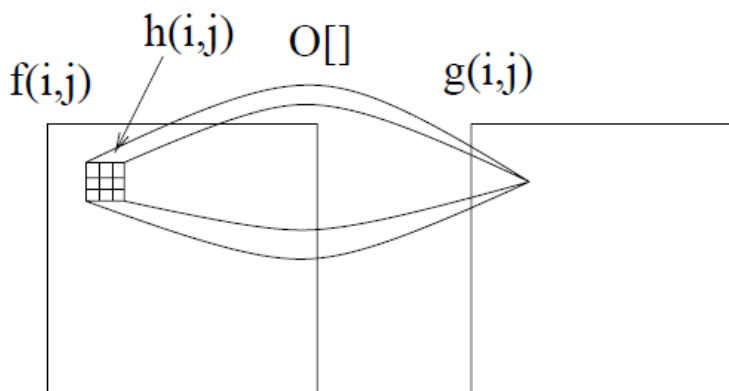
# Filtre neliniare în spațiul Real

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

Domeniul de definiție al măștii  $h(m, n)$  este dat de  $w$ .

Operația de filtrare este definită de masca  $h(i, j)$  și de operatorul  $O[]$ .

La majoritatea filtrelor neliniare avem:  $h(i, j) = 1 \quad i, j \in w$ ,  
operația de filtrare fiind controlată doar de operatorul  $O[]$





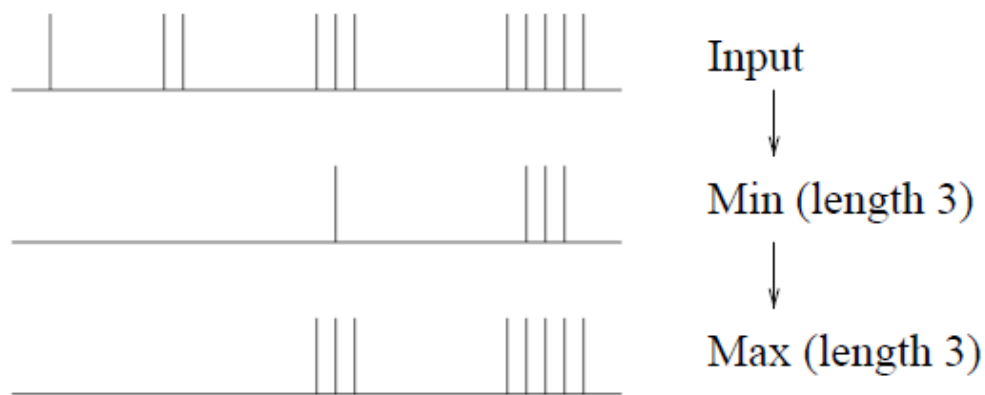
# Filtre “shrink & expand”

Shrink = îngustare; Expand = lărgire

$O[] = \text{Min}[] \Rightarrow$  shrink: obiectele luminoase (alb = obiect) sunt reduse în dimensiune cu un ordin de mărime egal cu dimensiunea filtrului

$O[] = \text{Max}[] \Rightarrow$  expand: obiectele luminoase sunt extinse în dimensiune cu un ordin de mărime egal cu dimensiunea filtrului

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



Obs: aplicarea filtrelor nu este comutativă:

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



## Filtre shrink/expand 2D

Exemplu: fereastră 1D de dimensiune 3 (din exemplul anterior)  $\Rightarrow$   
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  
4x4:



Input



Binary Threshold



Binary Shrink



Binary Expand



# Filtrul Threshold Average

Compară fiecare pixel cu media vecinilor (exclusiv valoarea lui) și netezește (înlocuiește pixelul curent) doar dacă diferența 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:  $h(i,j) = \begin{bmatrix} k & k & k \\ k & 0 & k \\ k & k & k \end{bmatrix} \quad k = 1/(M^2 - 1) = 0.125$

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

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



# Filtrul median

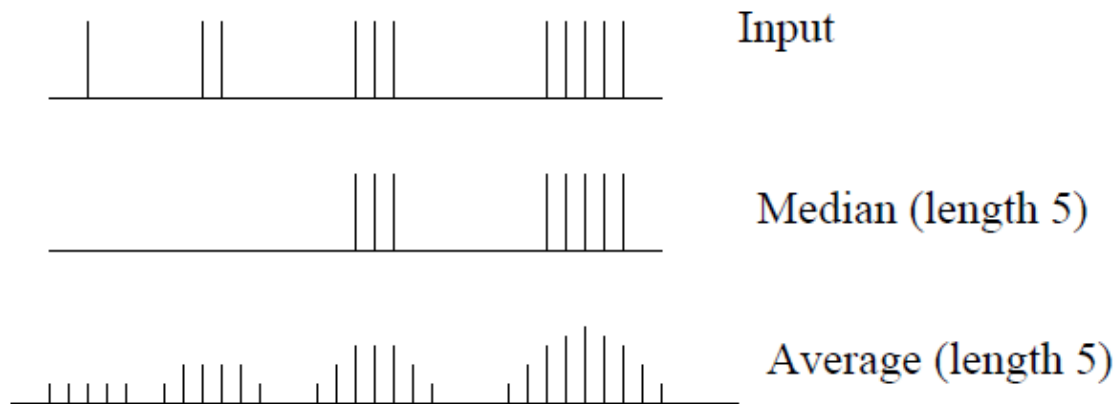
$$O[] = \text{Median}[]$$

Median := elementul de mijloc dintr-o mulțime sortată

Exemplu:

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

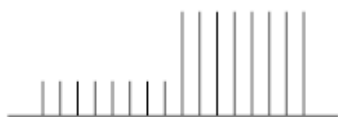
În cazul 1D filtrul median elimină toate trăsăturile/obiectele cu dimensiune mai mică decât  $M/2+1$ , păstrând toate celelalte trăsături



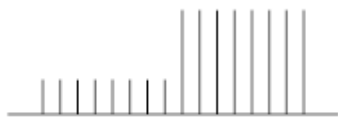


# Proprietăți (filtrul median)

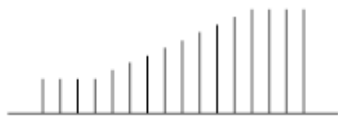
Netezire fără alterarea punctelor de muchie (low-pass selectiv):



Input



Median (any length)



Average (length 5)

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



$3 \times 3$  Median



$5 \times 5$  Median