

Concepte și aplicații în Vederea Artificială

Bogdan Alexe

bogdan.alexe@fmi.unibuc.ro

Cursul 4

anul III, Opțional Informatică, semestrul I, 2018-2019

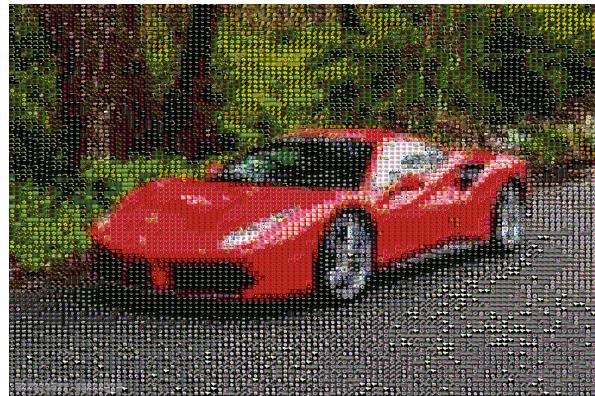
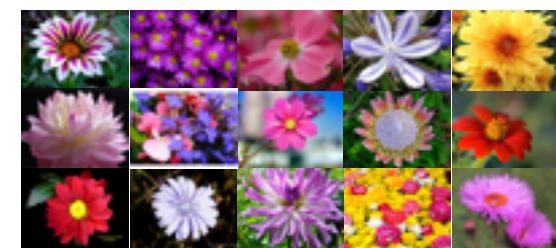
Tema 1 – deadline duminică

1.4 Predarea temei

Puneți într-o arhivă cu numele *tema1_cod.zip* codul vostru Matlab. Puneți într-un document cu numele *tema1_rezultate.pdf* următoarele:

- (3 puncte) mozaicurile obținute pentru imaginile din directorul '*../data/imaginiTest/*' la rularea algoritmului vostru pentru diferite valori ale parametrului *numarPieseMozaicOrizontala* (100, 75, 50, 25) și criteriul distanței euclidiene dintre culorile medii corespondând influență lor asupra rezultatelor obținute.
 - (1 punct) mozaicurile obținute pentru imaginile din directorul '*../data/imaginiTest/*' pentru modul de aranjare 'aleator' și criteriul distanței euclidiene dintre culorile medii. Pentru rezolvarea acestui punct trebuie să scrieți funcția *adaugăPieseMozaicModAleator.m*. Pentru imaginea inițială din Figura 1 ar trebui să obțineți ceva similar cu imaginile din Figura 2.
 - (1 punct) mozaicurile obținute pentru imaginile din directorul '*../data/imaginiTest/*' astfel: implementați o modificare a funcției *adaugăPieseMozaicPeCaraș.m* astfel încât mozaicul obținut să aibă proprietatea că nu există două piese adiacente (stânga, dreapta, jos, sus) identice. Pentru imaginea inițială din Figura 1 ar trebui să obțineți ceva similar cu imaginile din Figura 3.
 - (1 punct): înlocuiți colecția furnizată de noi (cu cele 500 de piese ce reprezintă flori) cu o colecție proprie. O posibilitate ar fi să downloadați setul de date CIFAR – 10 de la adresa <https://www.cs.toronto.edu/~kriz/cifar.html>. CIFAR – 10 conține 60000 de imagini color de dimensiuni 32 × 32 pixeli cu obiecte din 10 clase: avion, automobil, etc. Realizați mozaicuri tematice, construind mozaicuri pentru imagini conținând obiecte din aceste clase cu piesele corespunzătoare: realizați un mozaic pentru o imagine cu un automobil folosind piese cu automobile. Includeți în pdf-ul vostru cel puțin 5 exemple de mozaicuri tematice.
 - (1 punct): modificați proiectul vostru astfel încât mozaicurile obținute să aibă piese hexagonale. Includeți în pdf-ul vostru mozaicurile obținute astfel pentru imaginile din directorul '*../data/imaginiTest/*'. Pentru imaginea inițială din Figura 1 ar trebui să obțineți ceva similar cu imaginile din Figura 4.
 - (1 punct): modificați punctul anterior astfel încât mozaicurile obținute să aibă proprietatea că nu există două piese hexagonale adiacente (sus, jos, stânga sus, stânga jos, dreapta sus, dreapta jos) identice. Includeți în pdf-ul vostru mozaicurile obținute astfel pentru imaginile din directorul '*../data/imaginiTest/*'.
- Se va nota cu 1 punct prezentarea proiectului în format pdf. Vom lua în calcul aspecte precum: așezarea în pagină, comentariile ce însoțesc imaginile prezentate, exemplele alese.

Oficiu 1 punct.



Cursul trecut

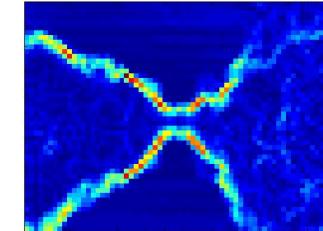
- Filtrarea liniară
 - corelație, convoluție
 - filtre: accentuare
- Filtrarea neliniară
 - filtrul median
- Aplicații:
 - reducerea zgomotului în imagini (filtrul median)
 - găsirea şabloanelor
 - redimensionarea imaginilor
 - extragerea informației (muchii, textură)

Cursul de azi

- Gradienți



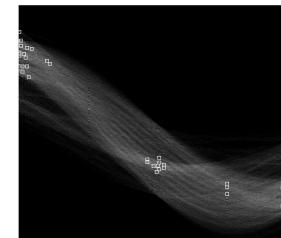
- Tema 2: redimensionarea imaginilor cu păstrarea conținutului



- Cum transformăm gradienții în muchii



- Aplicație: detectarea liniilor cu transformata Hough

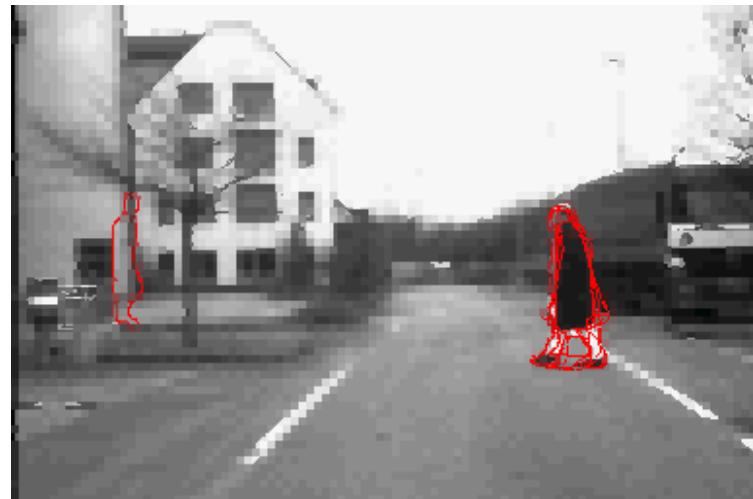
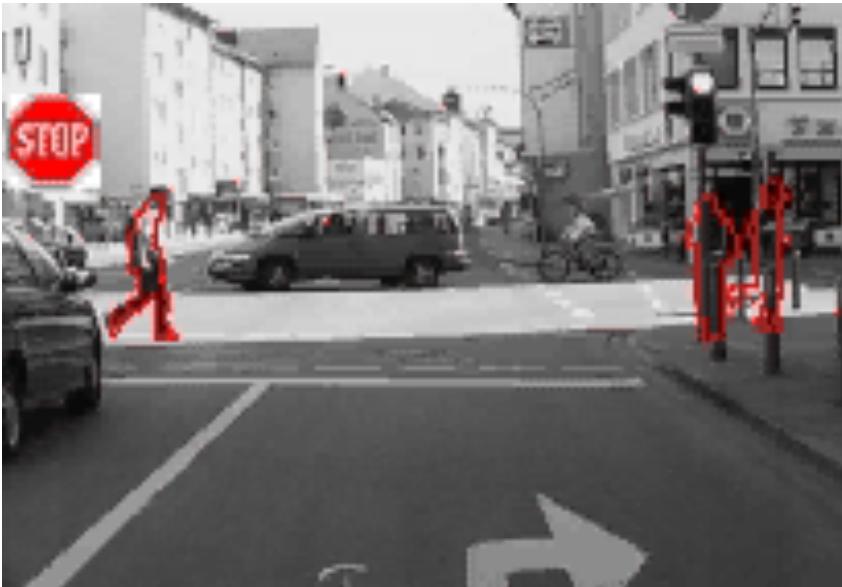


Aplicații – detectarea liniilor



https://www.youtube.com/watch?v=SFqAAseL_1g

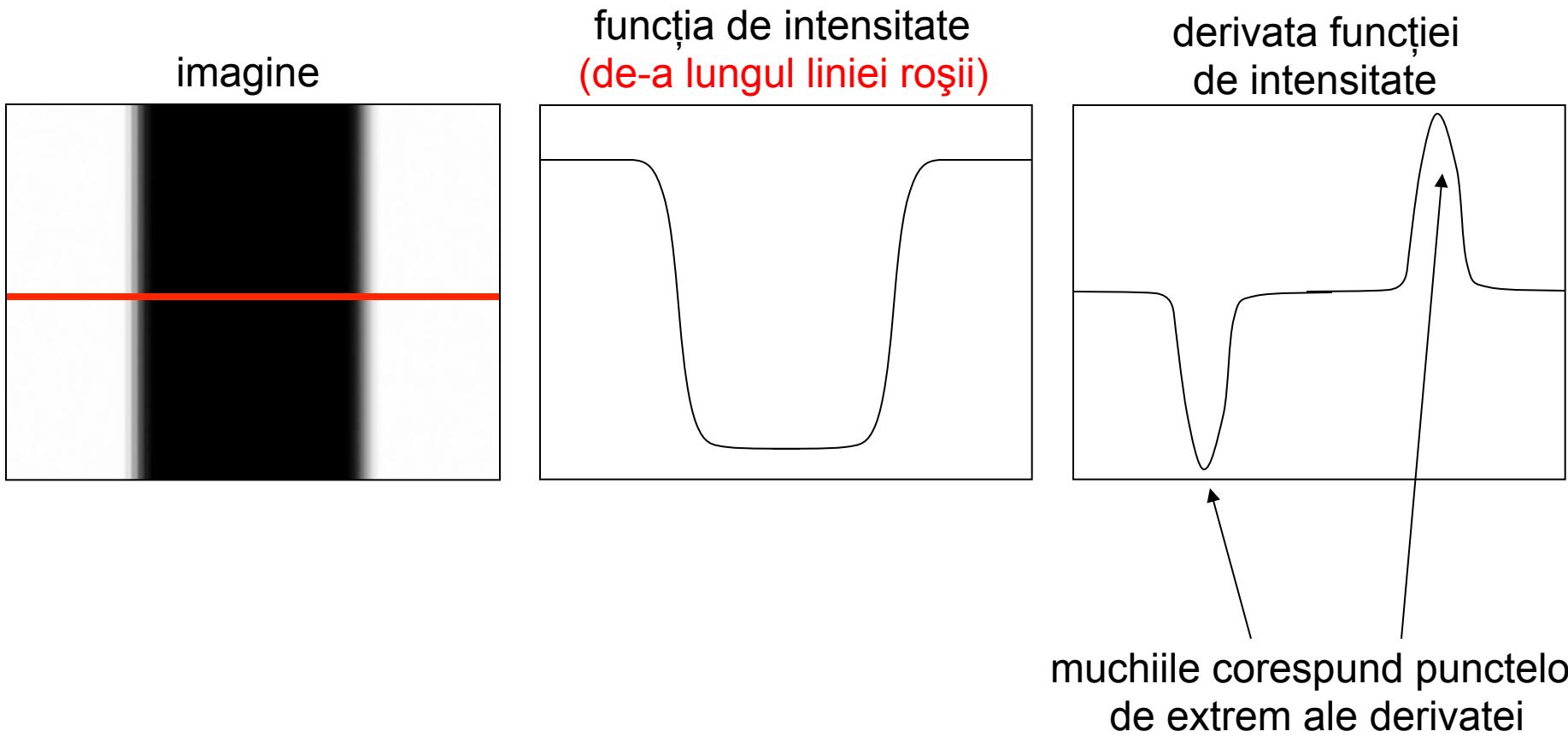
Aplicații: detectarea pietonilor pe baza contururilor (cursul următor)



Gradienti

Derivate și muchii

O muchie este locul în care se produce o schimbare bruscă a funcției de intensitate



Calculul derivatelor prin corelație/convoluție

Pentru o funcție $f(x,y)$, derivata parțială în raport cu x este:

$$\frac{\partial f(x, y)}{\partial x} = \lim_{\varepsilon \rightarrow 0} \frac{f(x + \varepsilon, y) - f(x, y)}{\varepsilon}$$

Pentru cazul discret (**cazul imaginilor**), putem aproxima derivata folosind diferențe finite:

$$\frac{\partial f(x, y)}{\partial x} \approx \frac{f(x + 1, y) - f(x, y)}{1}$$

Care este filtrul de corelație/convoluție care implementează relația de mai sus?

Filtre Sobel

Există aproximații mai bune pentru calcul derivatelor:

- filtre Sobel: calculează derivatele parțiale luând în considerare vecinătăți mai mari

-1	0	1
-2	0	2
-1	0	1

Filtru Sobel vertical pentru calculul derivatei parțiale

$$\frac{\partial f(x, y)}{\partial x}$$

1	2	1
0	0	0
-1	-2	-1

Filtru Sobel orizontal pentru calculul derivatei parțiale

$$\frac{\partial f(x, y)}{\partial y}$$

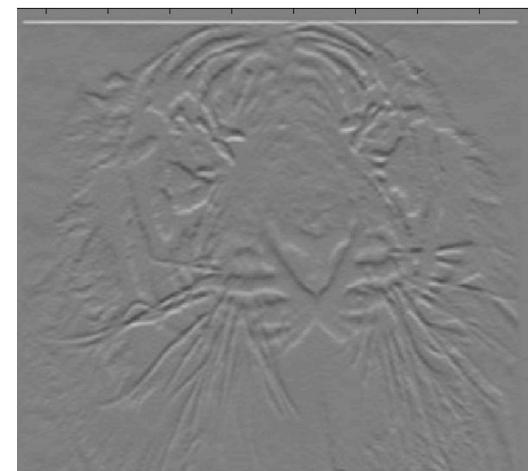
Alte filtre pentru diferențe finite

Prewitt: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

Sobel: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

Roberts: $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

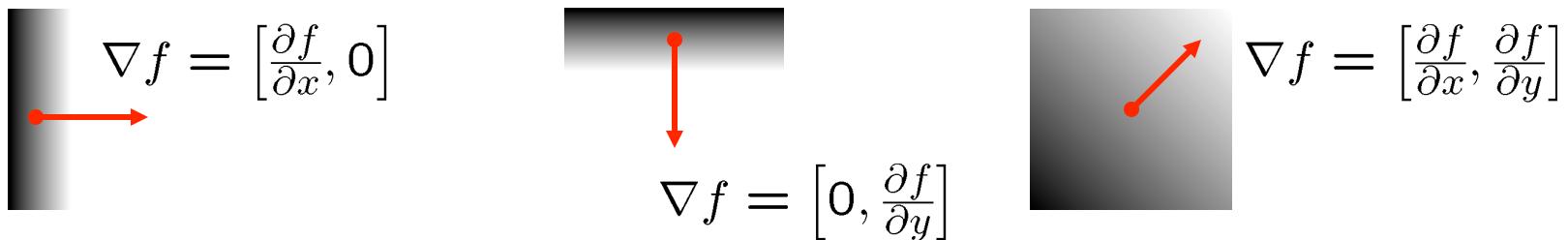
```
>> My = fspecial('sobel');  
>> outim = imfilter(double(im), My);  
>> imagesc(outim);  
>> colormap gray;
```



Gradientul unei imagini

Gradientul unei imagini:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$



Gradientul arată direcția celei mai rapide schimbări în intensitate

- Care este legătura dintre direcția gradientului și direcția muchiei?
- Direcția gradientului este perpendiculară pe direcția muchiei

Direcția gradientului este dată de: $\theta = \tan^{-1} \left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)$

Caracterizăm o muchie prin magnitudinea gradientului:

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \quad \text{sau} \quad \|\nabla f\| = \left| \frac{\partial f}{\partial x} \right| + \left| \frac{\partial f}{\partial y} \right|$$

Calculul gradientului unei imagini

1. calculez mai întâi derivatele parțiale în raport cu x și y

$$\frac{\partial f(x, y)}{\partial x} \quad \frac{\partial f(x, y)}{\partial y}$$

2. obțin magnitudinea gradientului:

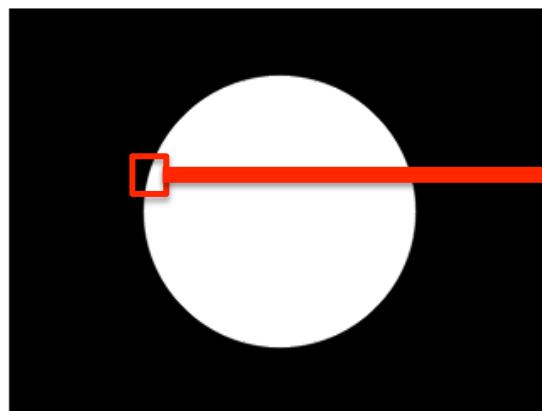
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \quad \text{sau} \quad \|\nabla f\| = \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right|$$

Calculul gradientului unei imagini

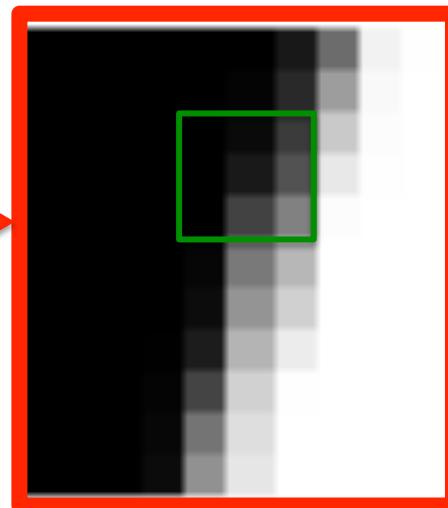
1. calculez mai întâi derivatata parțială în raport cu x $\frac{\partial f(x, y)}{\partial x}$

folosescul filtrul
Sobel vertical $M_x =$

$$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$



imagină f

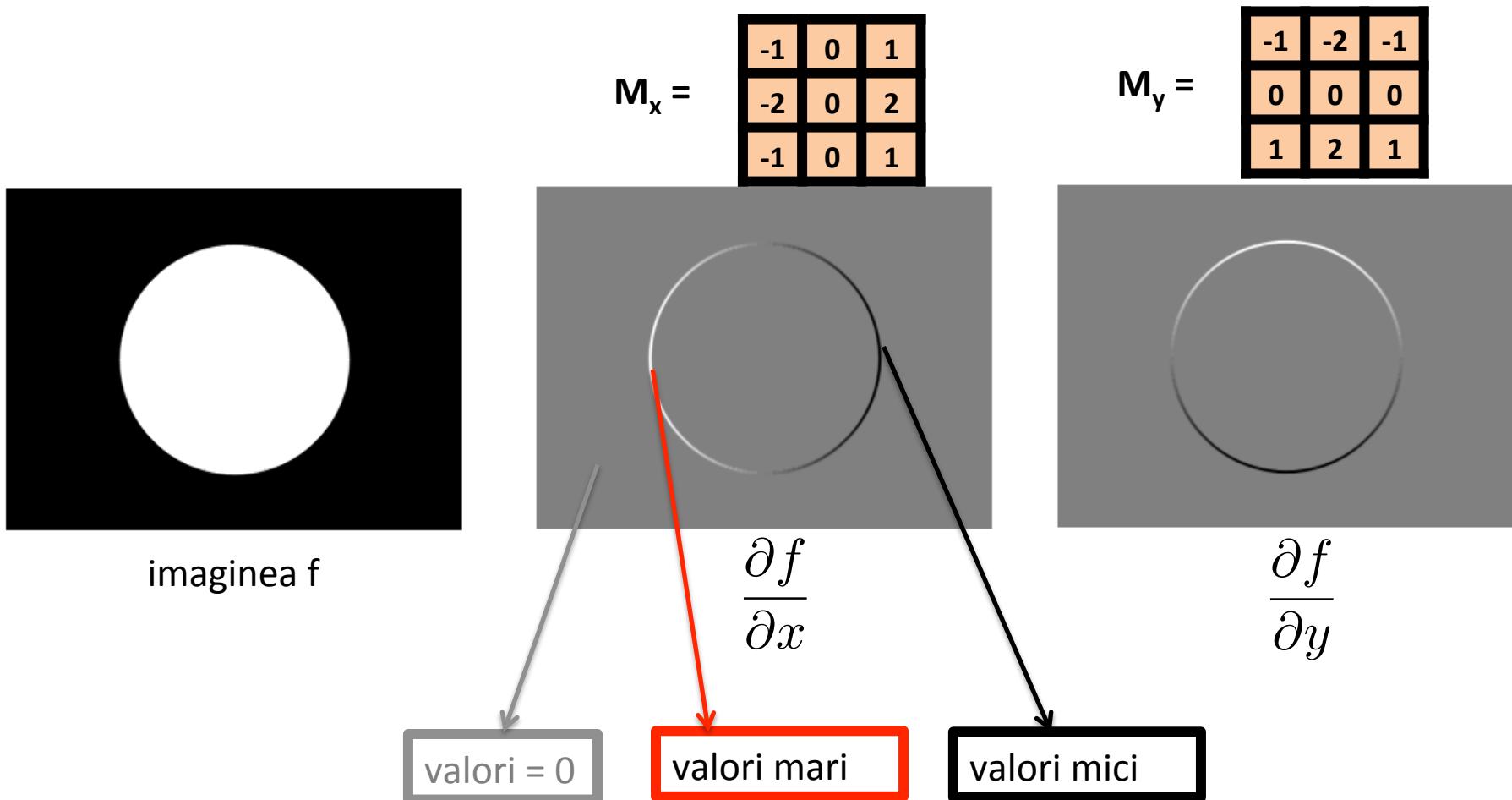


0	0	0	0	4	41	157	249	2
0	0	0	0	10	59	201	252	2
0	0	0	0	25	82	232	254	2
0	0	0	0	66	129	252	255	2
0	0	0	0	121	183	255	255	2
0	0	0	1	148	208	255	255	2
0	0	1	2	180	236	255	255	2
0	0	4	6	209	254	255	255	2
0	0	7	11	222	255	255	255	2

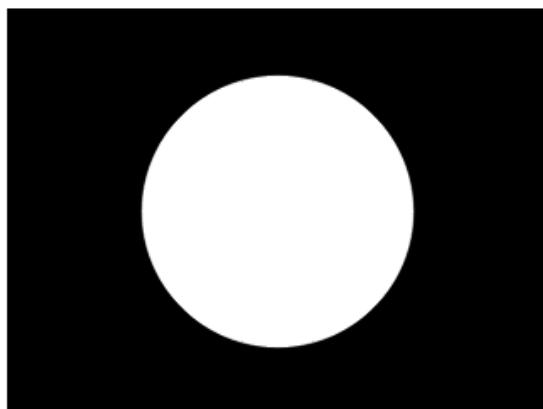


$$\begin{aligned} &= (-1) * 0 + 0 * 0 + 1 * 25 + \\ &+ (-2) * 0 + 0 * 0 + 2 * 66 + \\ &+ (-1) * 0 + 0 * 6 + 1 * 121 \\ &= 278 \end{aligned}$$

Calculul gradientului unei imagini



Calculul gradientului unei imagini



imagină f

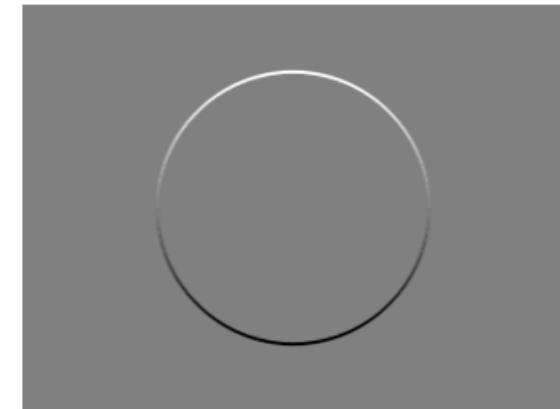


$$M_x =$$

-1	0	1
-2	0	2
-1	0	1

$$M_y =$$

-1	-2	-1
0	0	0
1	2	1

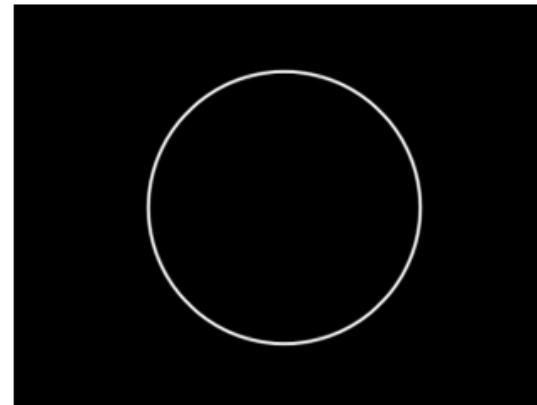


$$\frac{\partial f}{\partial x}$$

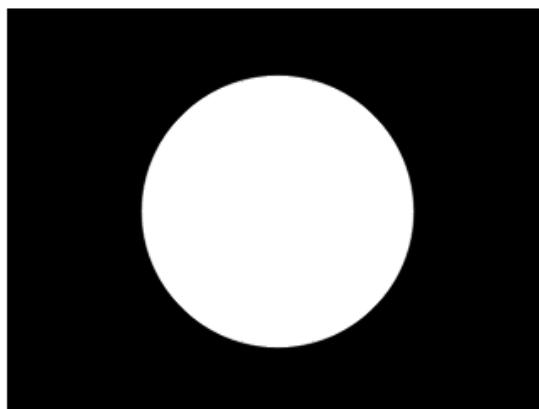
$$\frac{\partial f}{\partial y}$$

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Magnitudinea gradientului



Calculul gradientului unei imagini

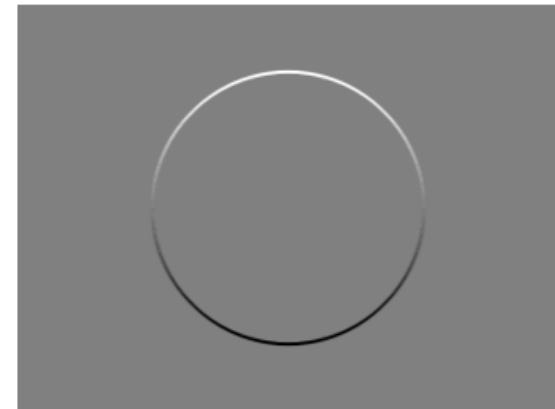


imagină f



$$M_x = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$

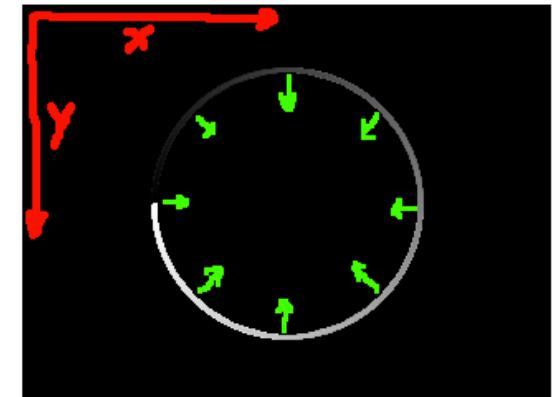
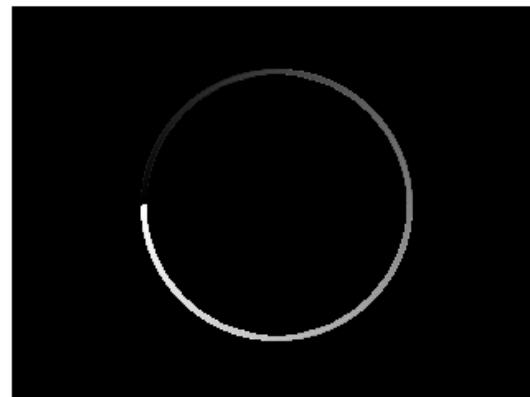
$$M_y = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$



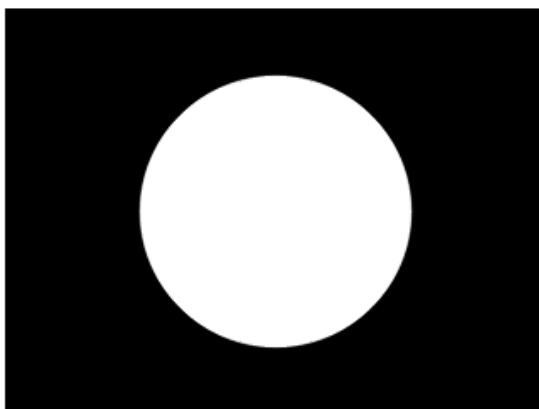
$$\frac{\partial f}{\partial x} \qquad \qquad \qquad \frac{\partial f}{\partial y}$$

$$\theta = \tan^{-1} \left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)$$

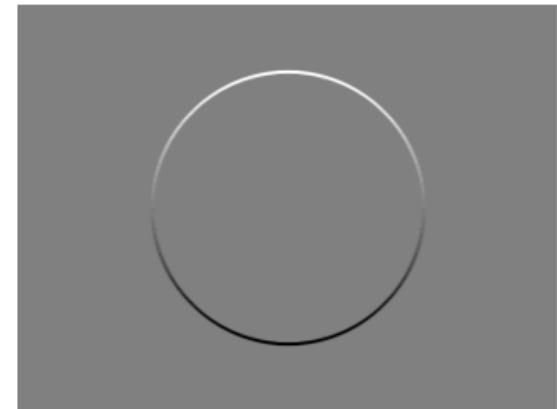
Orientarea gradientului



Calculul gradientului unei imagini



imagină f

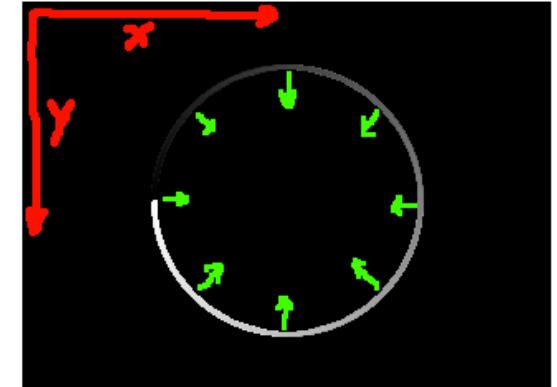
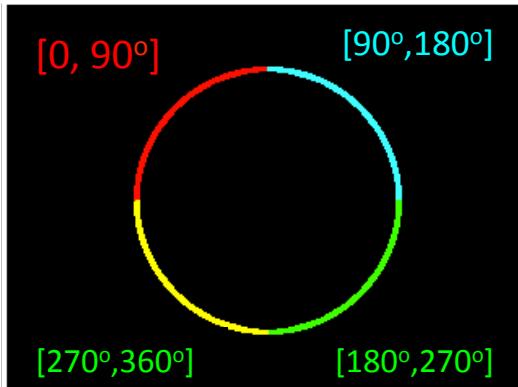


$$M_x = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$

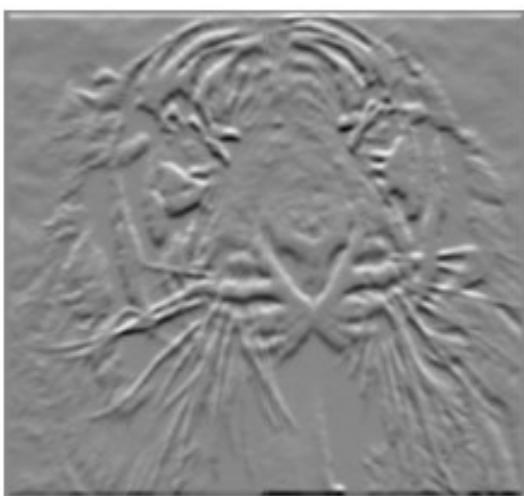
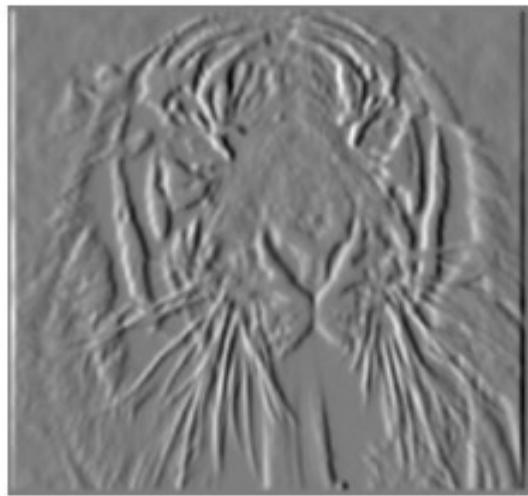
$$M_y = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$

$$\theta = \tan^{-1} \left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)$$

[0, 90°]
[90°, 180°]
[180°, 270°]
[270°, 360°]



Calculul gradientului unei imagini



$$\frac{\partial f(x, y)}{\partial x}$$

$$\frac{\partial f(x, y)}{\partial y}$$

$$\sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Magnitudinea gradientului

Calculul gradientului unei imagini



Imagine



Magnitudinea gradientului

Tema 2:

Redimensionarea imaginilor cu păstrarea conținutului

Redimensionarea imaginilor cu păstrarea conținutului

Imagine inițială



Redimensionare imagine: vreau să măresc/micșorez lătimea/înălțimea imaginii

Idee: adaug/elimin însiruirile de pixeli ce conectează extremitățile imaginii



Cum le aleg?

Redimensionarea imaginilor cu păstrarea conținutului



Redimensionare cu păstrarea conținutului



Redimensionare ușuală
(funcția **imresize** în MATLAB)

Ideea de bază



Shai Avidan
Mitsubishi Electric Research Lab
Ariel Shamir
The interdisciplinary Center & MERL

<https://www.youtube.com/watch?v=6NcIJXTlugc>

Ideea de bază



Redimensionare cu păstrarea conținutului

Intuiție:

- Păstrăm conținutul cel mai “interesant”
→ Preferăm să eliminăm/adăugăm pixeli cu gradient mic
- Pentru a reduce/mări o dimensiune , eliminăm/adăugăm însiruirile de pixeli (drumuri neregulate) ce conectează extremitățile
→ Soluție optimă folosind programarea dinamică

Ideea de bază – eliminare de drumuri



imaginea I

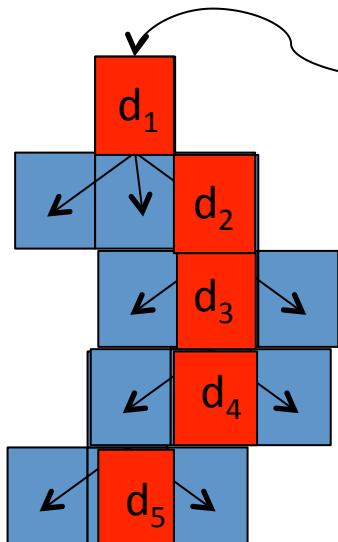


gradientul imaginii I

$$\nabla I = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$$

- Eliminăm ‘drumuri’ (înșiruiri de pixeli) fără a afecta vizibil imaginea:
 - măsurăm ‘costul unui drum’ ca suma magnitudinilor gradientilor pixelilor ce alcătuiesc drumul
- La fiecare iterație, alegem drumul cu **costul cel mai mic** din imagine

Algoritmul



imagină I



gradientul imaginii I

$$\nabla I = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$$

Fie **d** un **drum vertical** format din N pixeli: $\mathbf{d} = (d_1, d_2, \dots, d_N)$

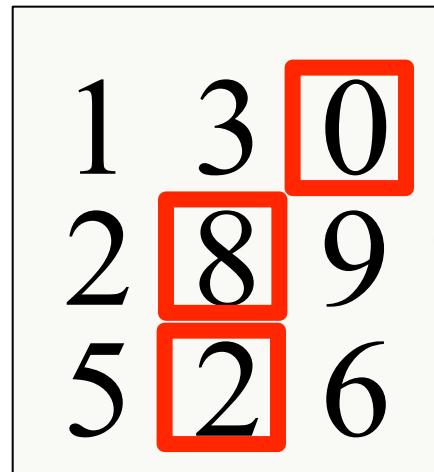
Definim **costul unui drum** : $Cost(\mathbf{d}) = \sum_{i=1}^N \nabla I(d_i)$

Drumul optim minimizează acest cost: $\mathbf{d}^* = \min_{\mathbf{d}} Cost(\mathbf{d})$

Calculăm **drumul optim** în mod eficient folosind **programarea dinamică**

Cum identicăm drumul de cost minim?

- Mai întâi, considerăm o strategie **greedy**:



La fiecare pas aleg cea mai bună soluție locală (aleg pixelii cu gradientul cel mai mic)

Combinarea optimelor locale NU conduce la optim global

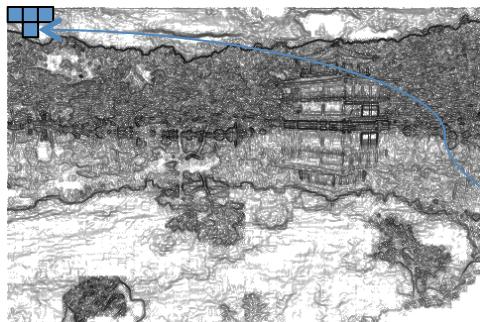


Magnitudinea gradientului

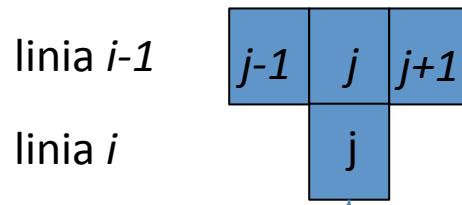
Algoritmul

- Calculăm pentru fiecare pixel (i,j) drumul de cost minim până la (i,j) cu ajutorul celor trei vecini:

$$\mathbf{M}(i, j) = MagGradient(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$



Magnitudinea gradientului



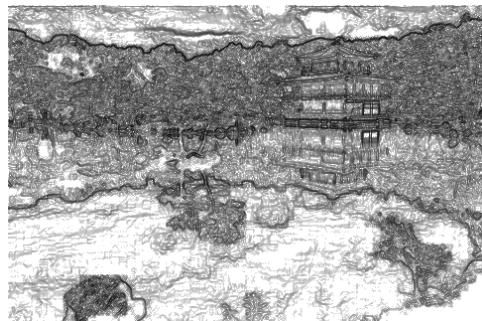
Matricea \mathbf{M} de costuri ale drumurilor
(aici pentru drumuri verticale)

- Valoarea minimă din ultima linie din \mathbf{M} reprezintă valoarea drumului vertical de cost minim. Poziția costului minim pe ultima linie localizează ultimul pixel din drum.
- Găsim drumul de cost minim mergând înapoi și găsind drumul minim cu ajutorul celor 3 vecini de sus din \mathbf{M}

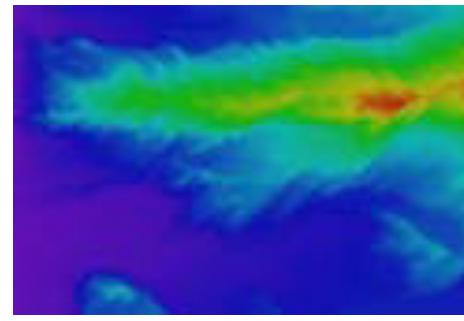
Exemplu – programare dinamică

$$\mathbf{M}(i, j) = MagGradient(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

1	3	0
2	8	9
5	2	6



Magnitudinea gradientului



Matricea M de costuri ale drumurilor
(aici pentru drumuri verticale)

Exemplu – programare dinamică

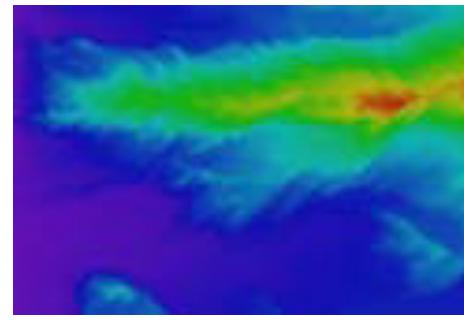
$$\mathbf{M}(i, j) = MagGradient(i, j) + \min(\mathbf{M}(i - 1, j - 1), \mathbf{M}(i - 1, j), \mathbf{M}(i - 1, j + 1))$$

1	3	0
2	8	9
5	2	6

1	3	0
3	8	9
8	5	14



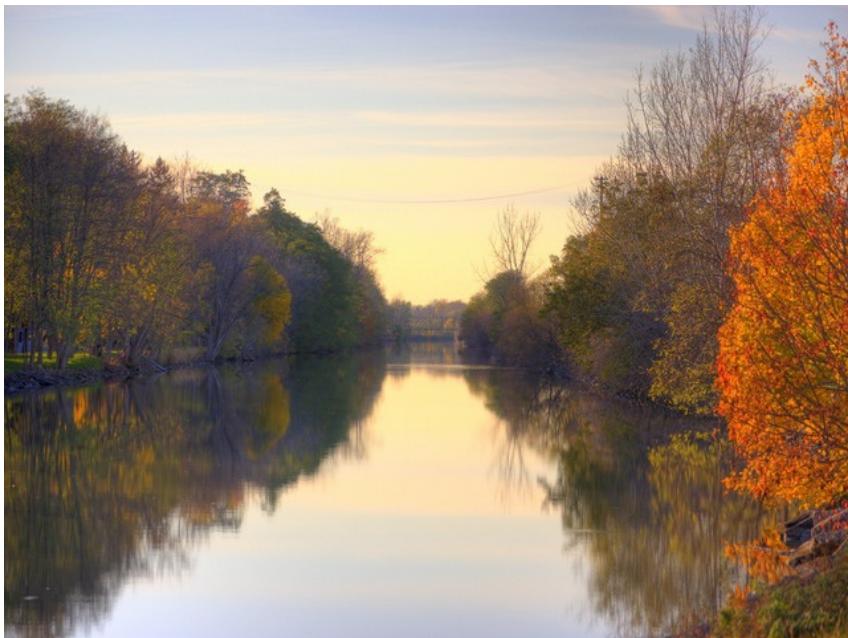
Magnitudinea gradientului



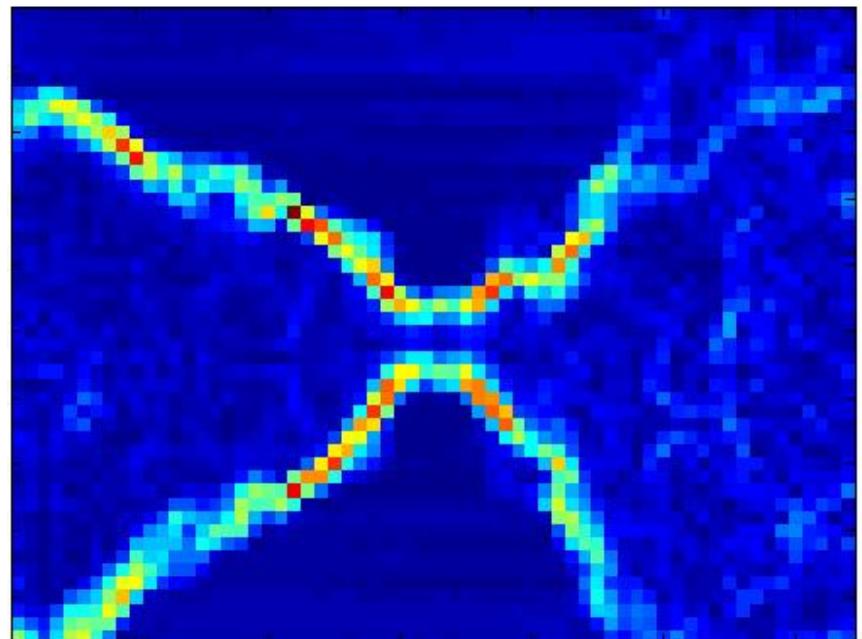
Matricea M de costuri ale drumurilor
(aici pentru drumuri verticale)

Exemplu

Imagine inițială

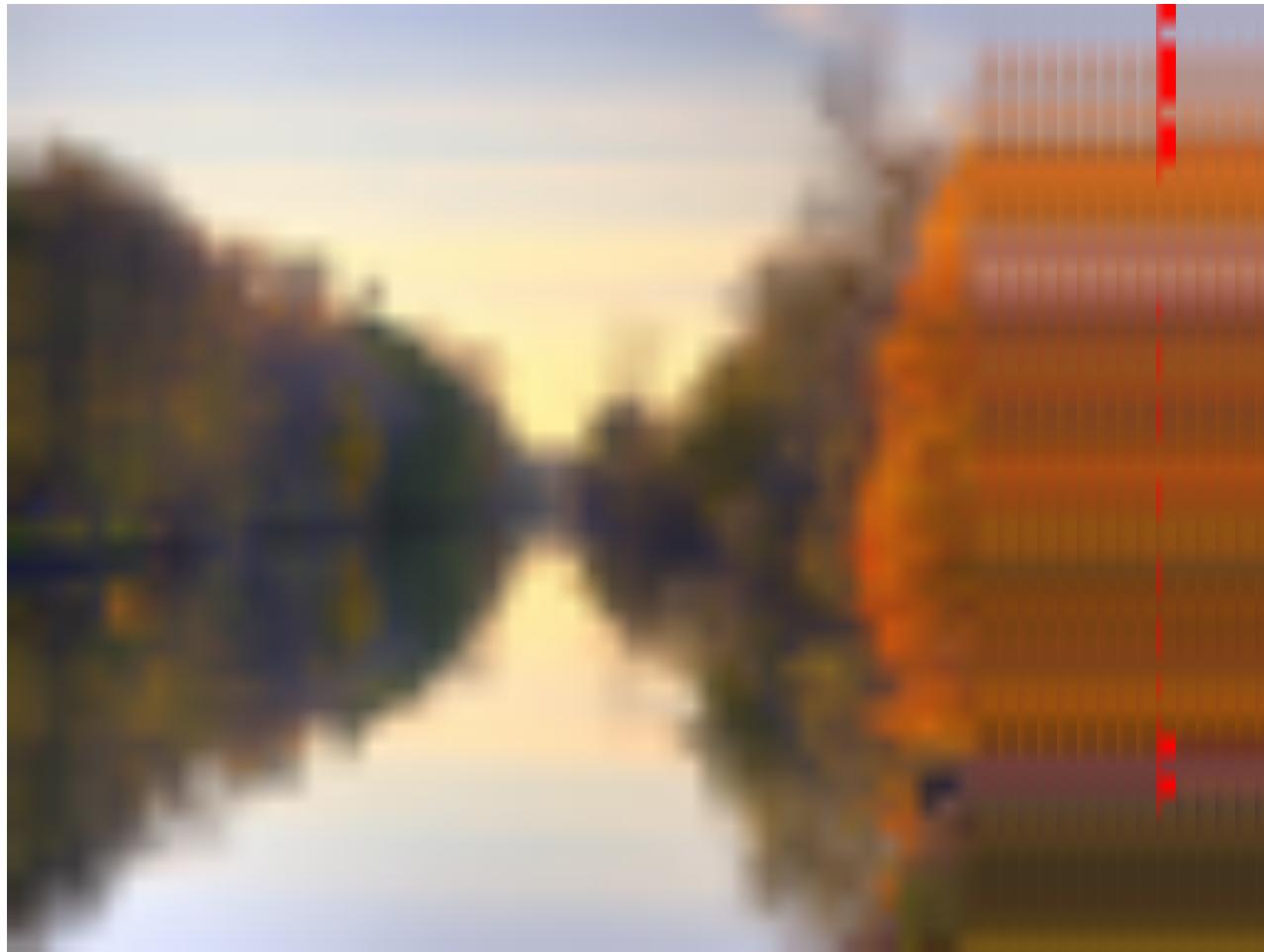


Matricea \mathbf{M} de costuri ale drumurilor



Albastru = cost mic
Roșu = cost mare

Exemplu



Rezultate

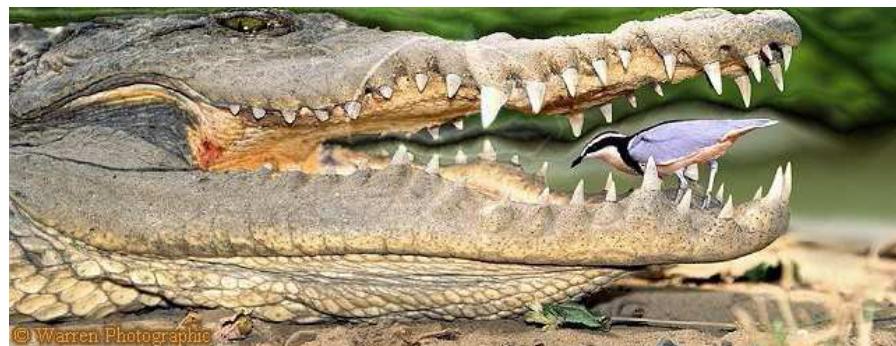


**Redimensionare cu
păstrarea conținutului**



**Redimensionare
uzuală (imresize)**

Rezultate



**Redimensionare cu
păstrarea conținutului**

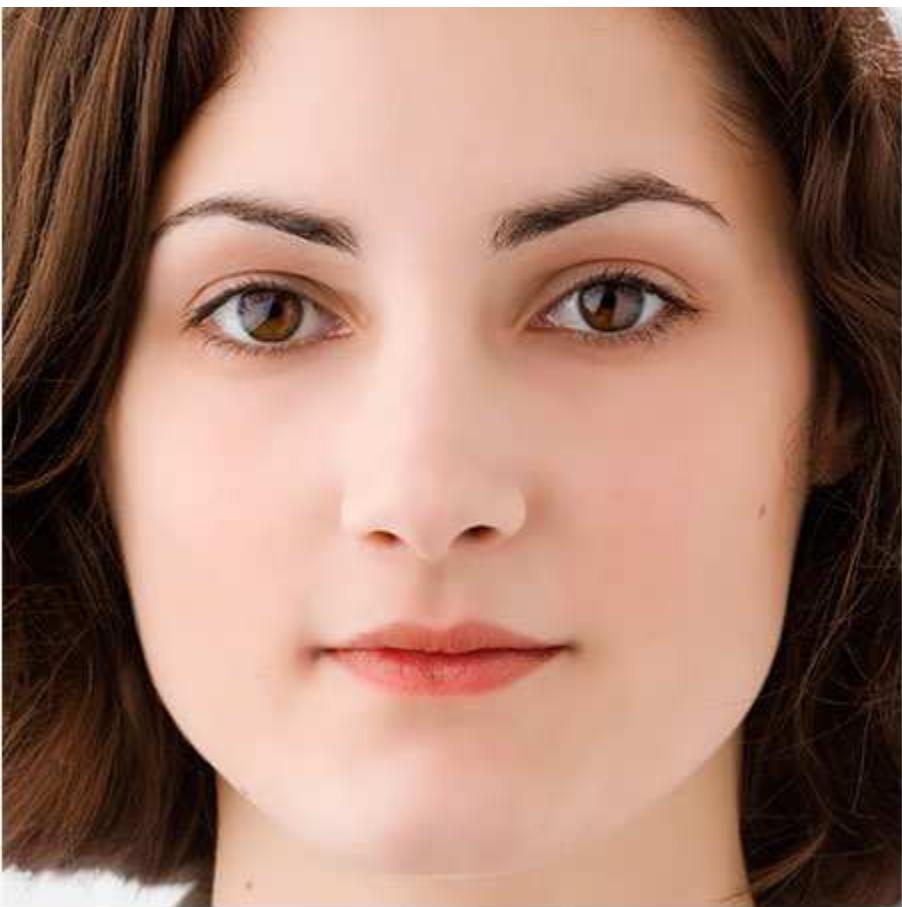


**Redimensionare
uzuală (imresize)**

Rezultate - failures



Rezultate - failures



Eliminarea unei regiuni din imagine



Demo Matlab

Cum transformăm
gradientii în muchii?

Funcția ‘edge’ în MATLAB

Găsește muchii după 6 metode:

1. metoda Sobel
2. metoda Prewitt
3. metoda Roberts
4. metoda Laplacian-ului unei funcții Gaussiene
5. metoda “zero-crossings”
6. metoda Canny

Metodele Sobel + Roberts + Prewitt

Găsește muchii ca punctele unde gradientul are valori mari (mai mari decât un prag). Calculează gradientul pe baza filtrelor corespunzătoare (Sobel, Roberts, Prewitt).

Prewitt: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

Sobel: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

Roberts: $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

Metodele Sobel + Roberts + Prewitt

Găsește muchii ca punctele unde gradientul are valori mari (mai mari decât un prag). Calculează gradientul pe baza filtrelor corespunzătoare (Sobel, Roberts, Prewitt).



Gradientul imaginii
calculat prin metoda Sobel



Gradientul imaginii calculat
prin metoda Roberts



Gradientul imaginii calculat
prin metoda Prewitt

Metodele Sobel + Roberts + Prewitt

Găsește muchii ca punctele unde gradientul are valori mari (mai mari decât un prag). Calculează gradientul pe baza filtrelor corespunzătoare (Sobel, Roberts, Prewitt).



```
E = edge(imgGray,'sobel')  
figure,imshow(E)
```



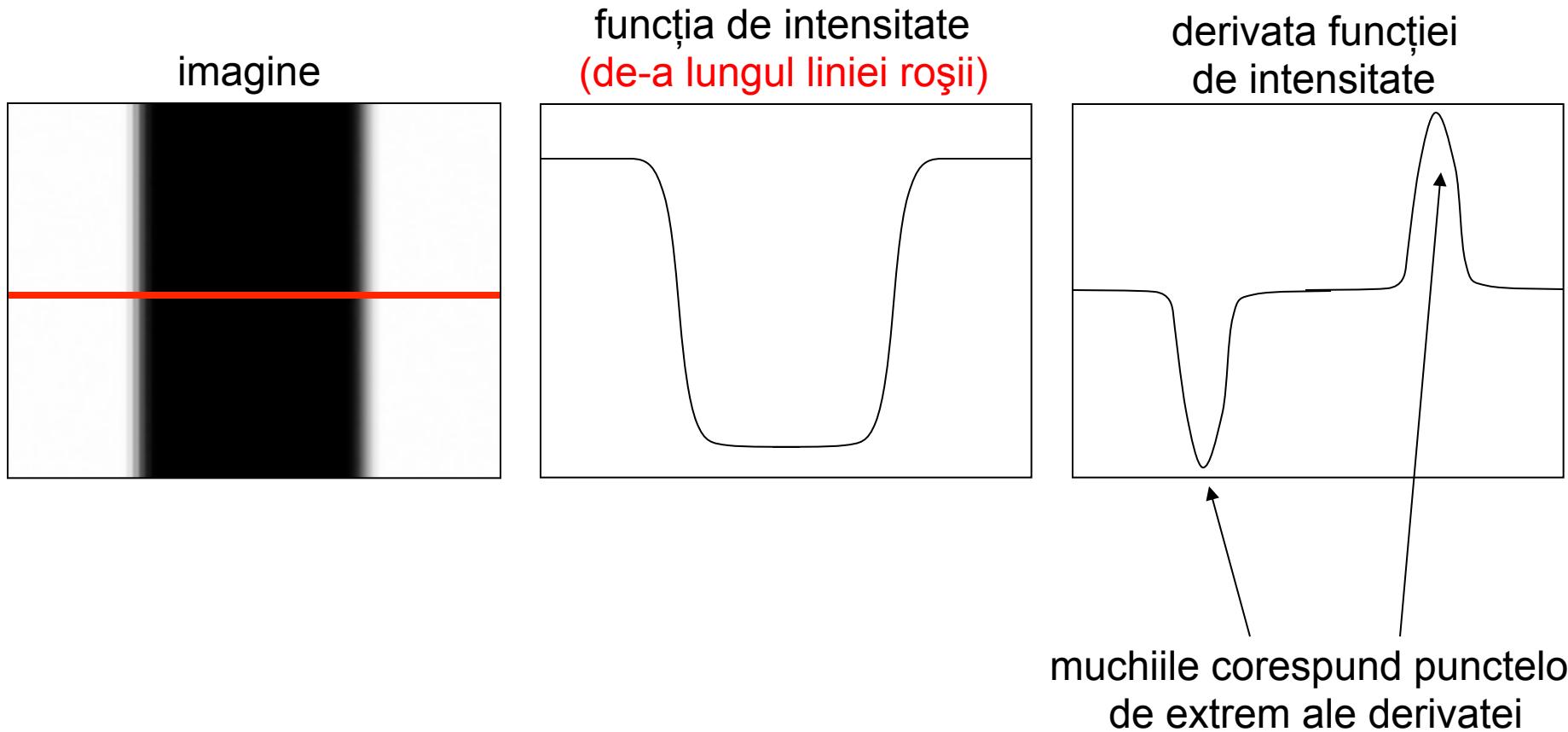
```
E = edge(imgGray,'roberts'); E = edge(imgGray,'prewitt');  
figure,imshow(E) figure,imshow(E)
```



Ahem nevoie de un prag = threshold pentru a obține muchii (pixelii din muchie = edgels). Toți pixelii cu gradient > prag devin 1 (edgels – parte din muchie), restul devin 0.

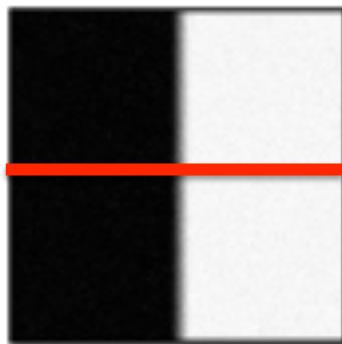
Obținerea filtrului Laplacian

O muchie este locul în care se produce o schimbare bruscă a funcției de intensitate

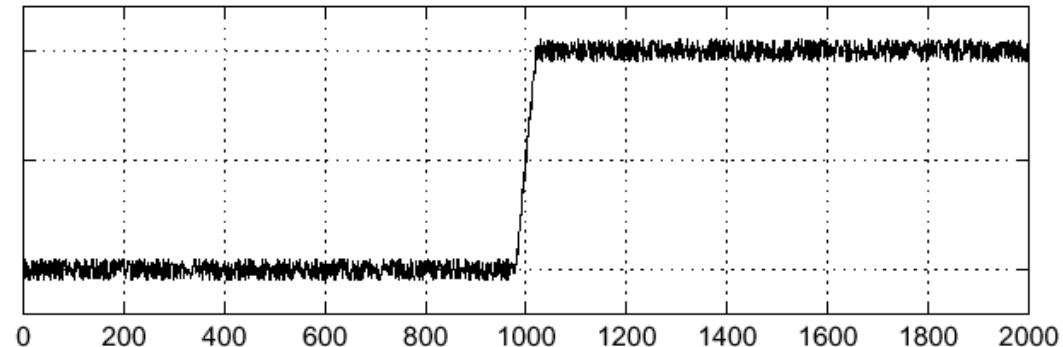


Efectul zgomotului – 1D

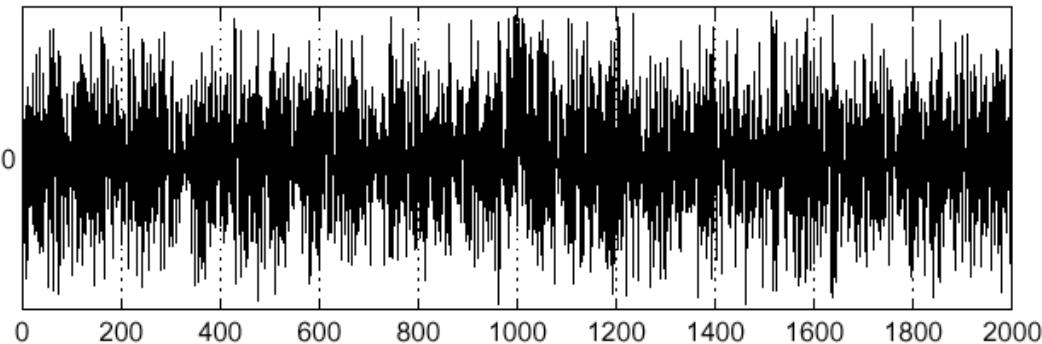
Considerăm linia roșie a imaginii
– plotăm intensitatea în funcție de poziție



$$f(x)$$

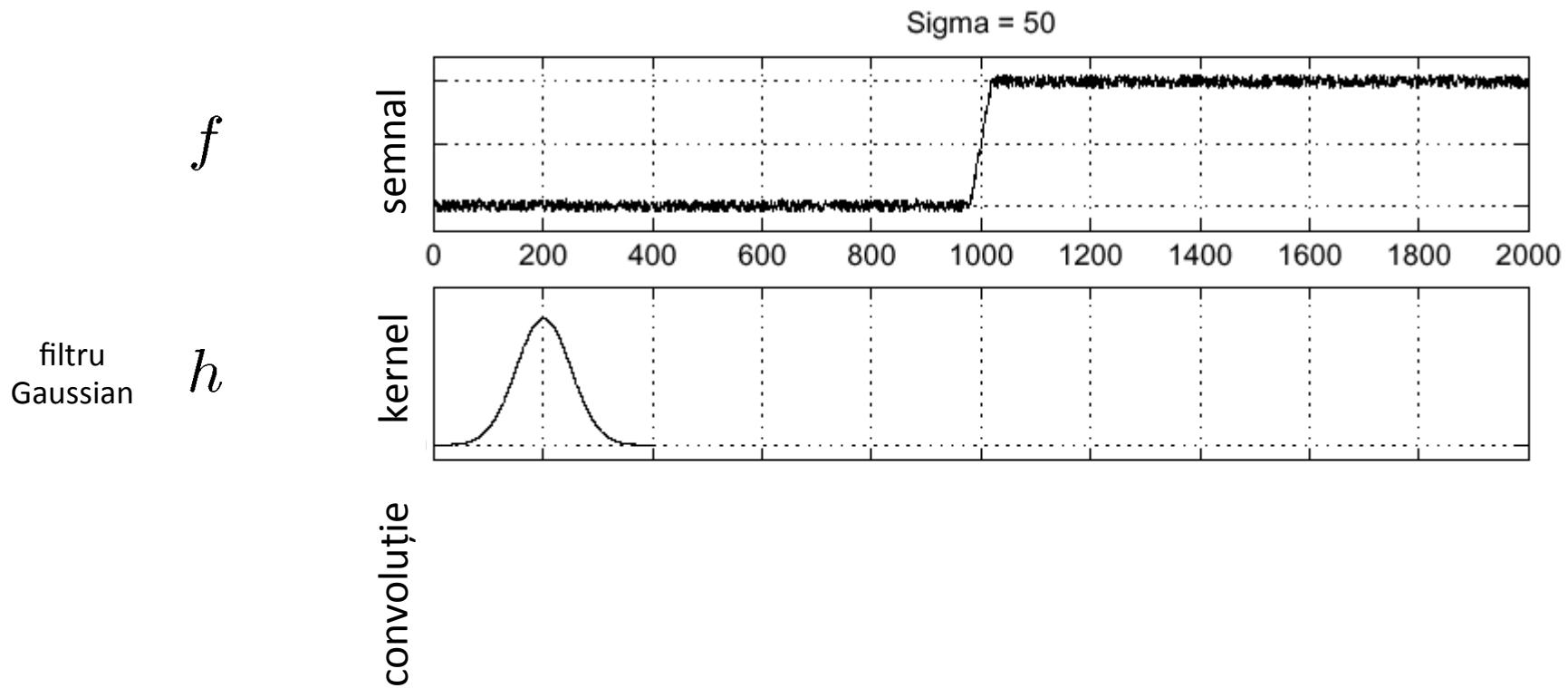


$$\frac{d}{dx}f(x)$$



Unde este muchia?

Soluție: blurăm cu un filtru Gaussian – 1D



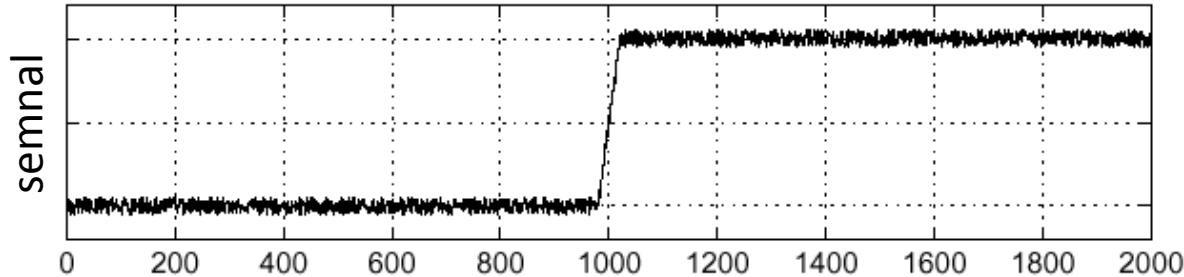
Unde este muchia? Căutăm punctele de maxim local în $\frac{\partial}{\partial x}(h \star f)$ = zerourile funcției $\frac{\partial^2}{\partial x^2}(h \star f)$

Proprietatea de diferențiabilitate a convoluției

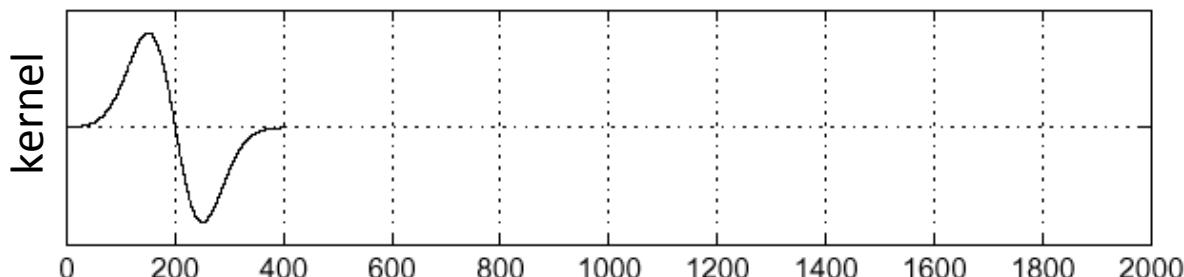
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

Sigma = 50

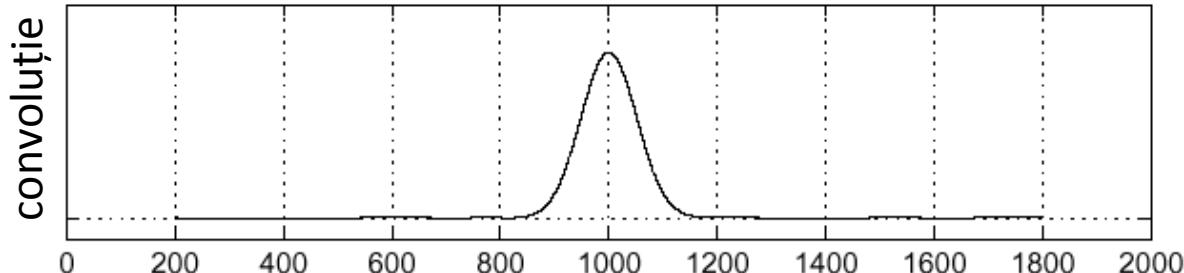
f



$\frac{\partial}{\partial x}h$

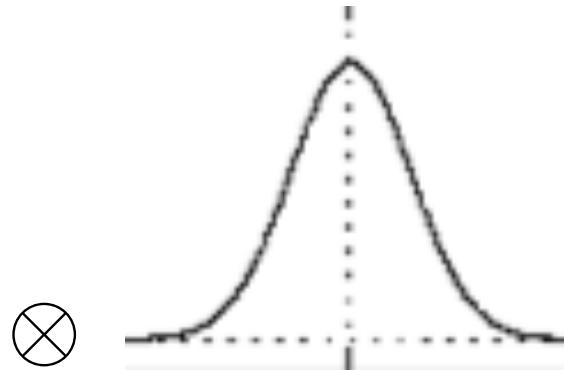


$(\frac{\partial}{\partial x}h) \star f$

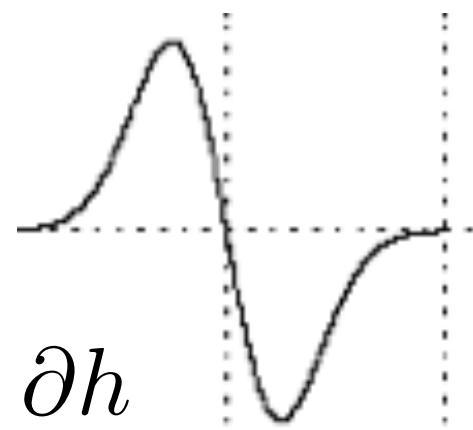


1D

-1	1
----	---



=



d_x

h

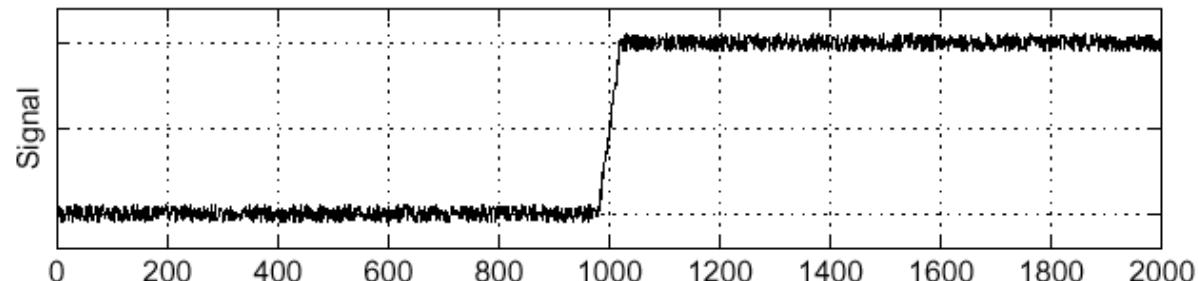
$\frac{\partial h}{\partial x}$

Laplacian-ul unei funcții Gaussiene – 1D

Considerăm $\frac{\partial^2}{\partial x^2}(h * f)$

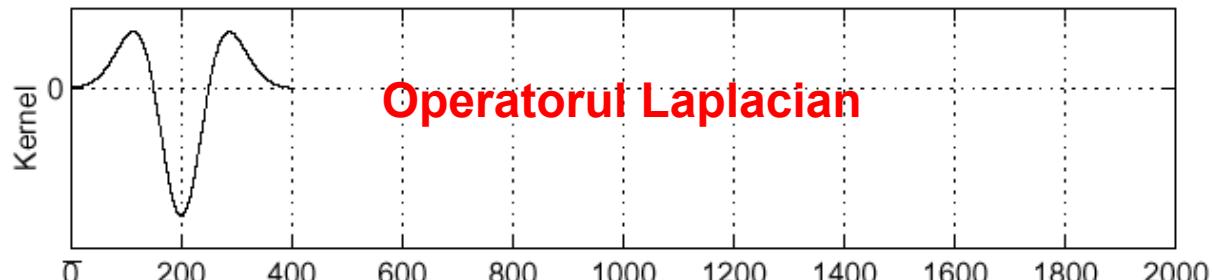
Sigma = 50

f

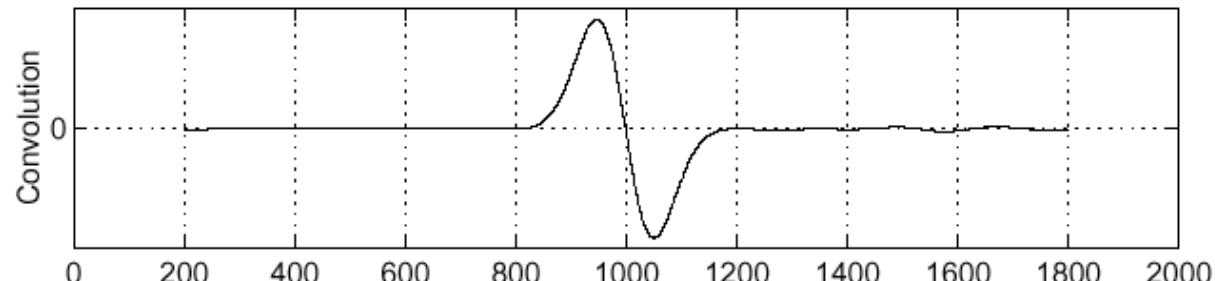


$\frac{\partial^2}{\partial x^2} h$

Operatorul Laplacian



$(\frac{\partial^2}{\partial x^2} h) * f$



Unde este muchia?

Zero-crossing ale funcției $(\frac{\partial^2}{\partial x^2} h) * f$

Slide adaptat după Steve Seitz

Derivata filtrelor Gaussiane - 2D

$$d_x \otimes (h \otimes I) = (d_x \otimes h) \otimes I$$

filtru de derivare

filtru Gaussian

imagină

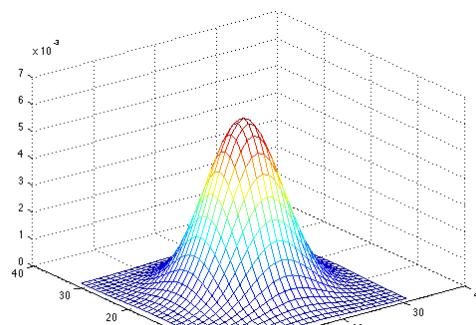
0	0	0
0	-1	1
0	0	0



$$\left[\begin{array}{ccccc} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{array} \right]$$

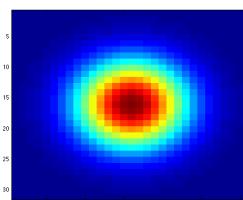


0	0	0
0	-1	1
0	0	0



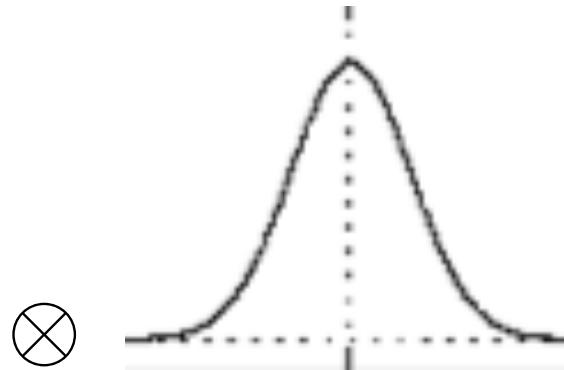
=

?

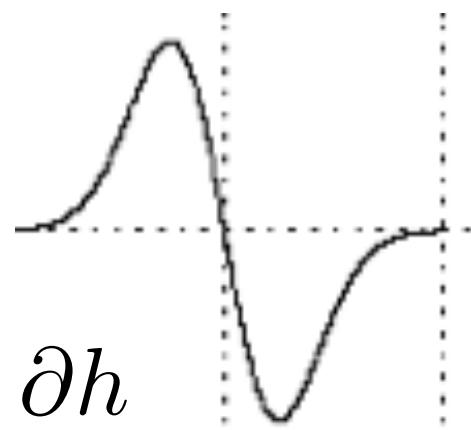


1D

-1	1
----	---



=



d_x

h

$\frac{\partial h}{\partial x}$

Derivata filtrelor Gaussiane - 2D

$$d_x \otimes (h \otimes I) = (d_x \otimes h) \otimes I$$

filtru de derivare

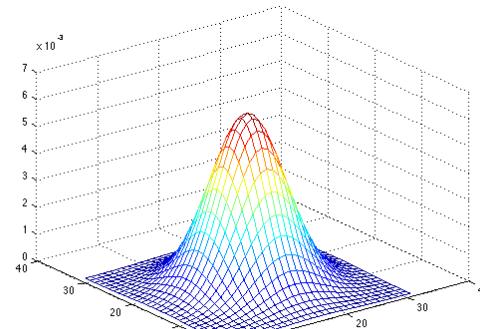
filtru Gaussian

imagină

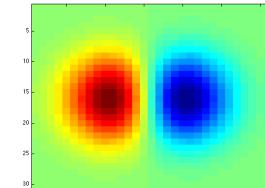
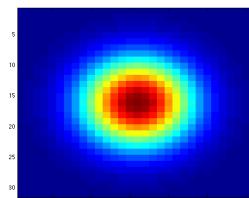
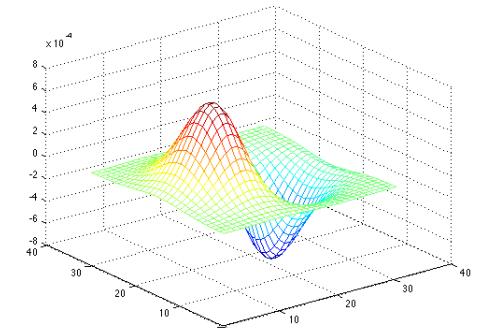
0	0	0
0	-1	1
0	0	0

$$\otimes \quad \left[\begin{array}{ccccc} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{array} \right]$$

0	0	0
0	-1	1
0	0	0



=



Derivata filtrelor Gaussiane - 2D

$$d_y \otimes (h \otimes I) = (d_y \otimes h) \otimes I$$

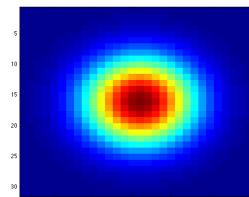
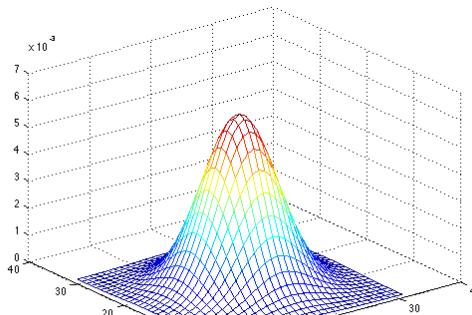
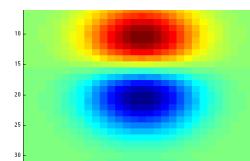
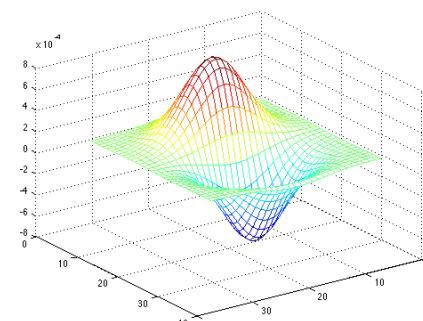
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

 \otimes

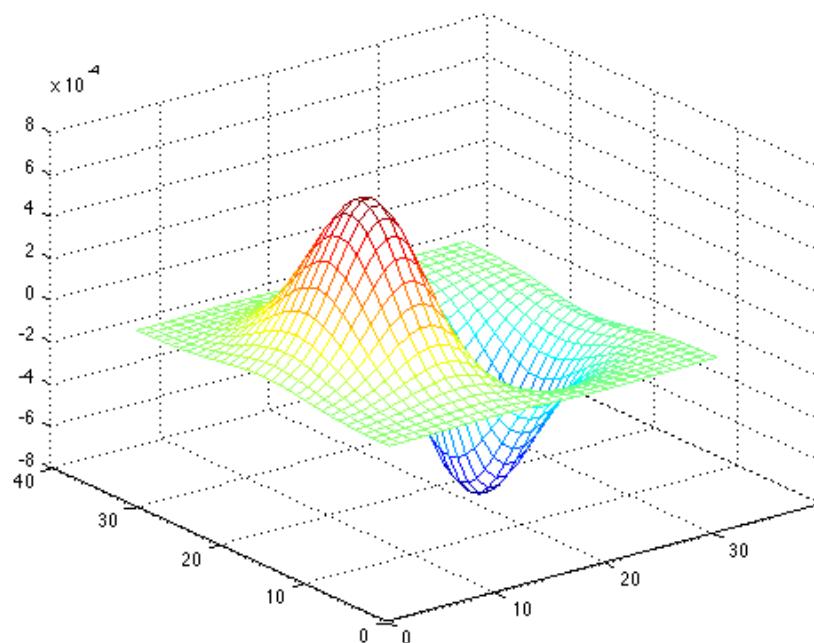
$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix}$$

 $]$

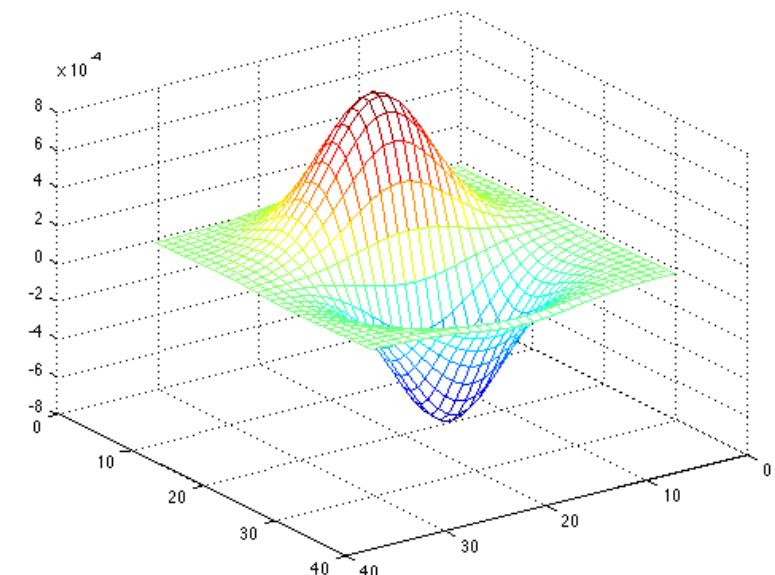
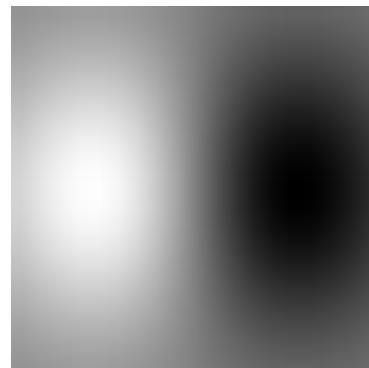
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

 \otimes  $=$ 

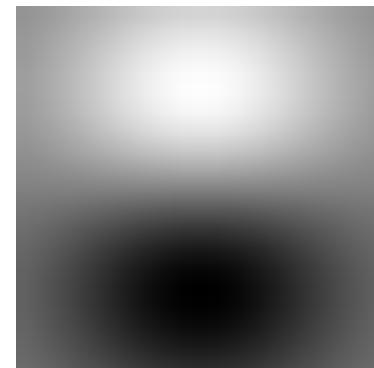
Derivata filtrelor Gaussiane - 2D



direcția x

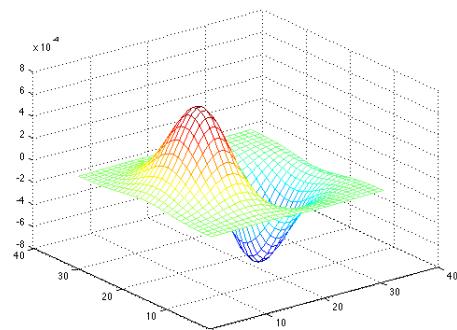


direcția y

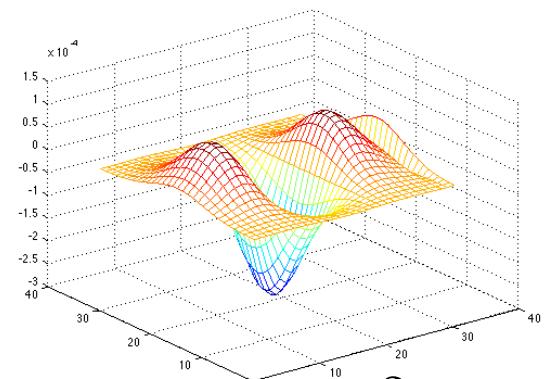


Derivata a doua a filtrelor Gaussiane

0	0	0
0	-1	1
0	0	0



=

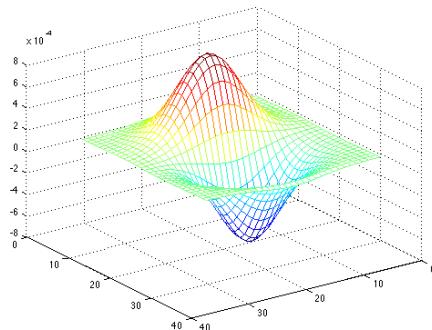


direcția x

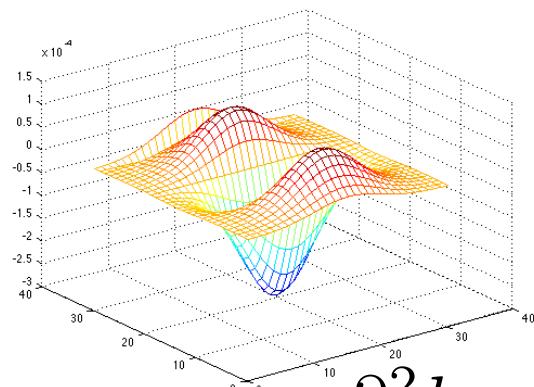
$$\frac{\partial h}{\partial x}$$

$$\frac{\partial^2 h}{\partial x^2}$$

0	0	0
0	-1	0
0	1	0



=



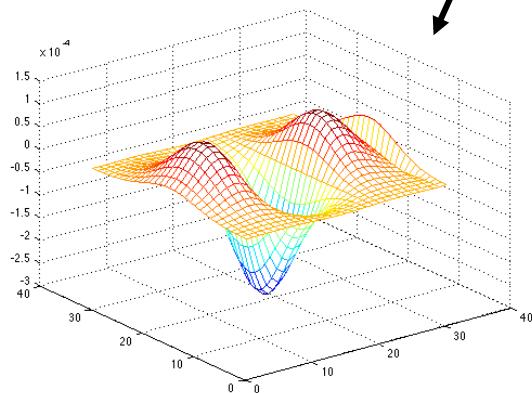
direcția y

$$\frac{\partial h}{\partial y}$$

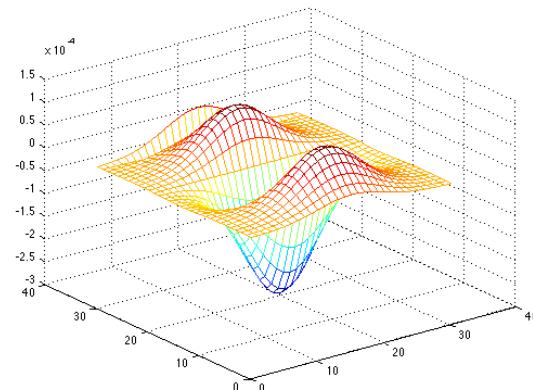
$$\frac{\partial^2 h}{\partial y^2}$$

Laplacian-ul unei funcții Gaussiene – 2D

$$\nabla^2 h = \frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2}$$

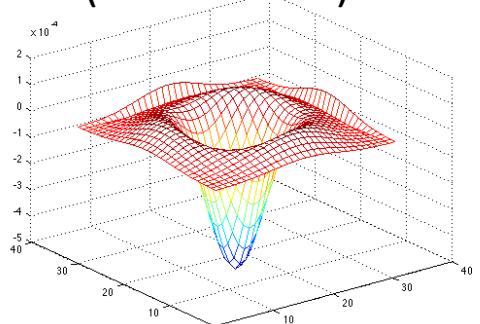


+



=

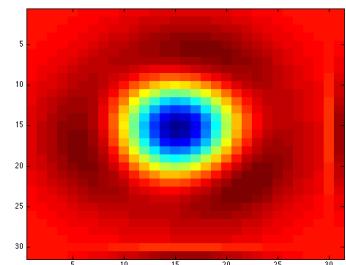
Forma generală a
unui filtru Laplacian
(mexican hat)



0	1	0
1	-4	1
0	1	0

MATLAB: `h = fspecial('log', size, sigma)`

Exemplu particular
de filtru Laplacian



Metoda Laplacianul unei funcții Gaussiene



```
E = edge(imgGray,'sobel');  
figure,imshow(E)
```



```
E = edge(imgGray,'roberts');  
figure,imshow(E)
```



```
E = edge(imgGray,'prewitt');  
figure,imshow(E)
```



```
E = edge(imgGray,'log');  
figure,imshow(E)
```

Proprietățile filtrelor

- Filtre de blurare (cursul trecut)
 1. valori pozitive;
 2. suma lor $= 1 \rightarrow$ pentru regiuni constante, output = input;
 3. gradul de blurare proporțional cu dimensiunea filtrului;
 4. elimină regiunile de pixeli cu varianță mare în intensitate (“high-frequency”); se mai numesc filtre trece-jos (“low-pass”)
- Filtre pentru detectarea muchiilor
 1. valori opuse pentru a avea un răspuns mare în regiuni cu contrast mare
 2. suma lor $= 0 \rightarrow$ pentru regiuni constante, răspuns = 0;
 3. cele mai mari valori în punctele de contrast maxim

Exemplu



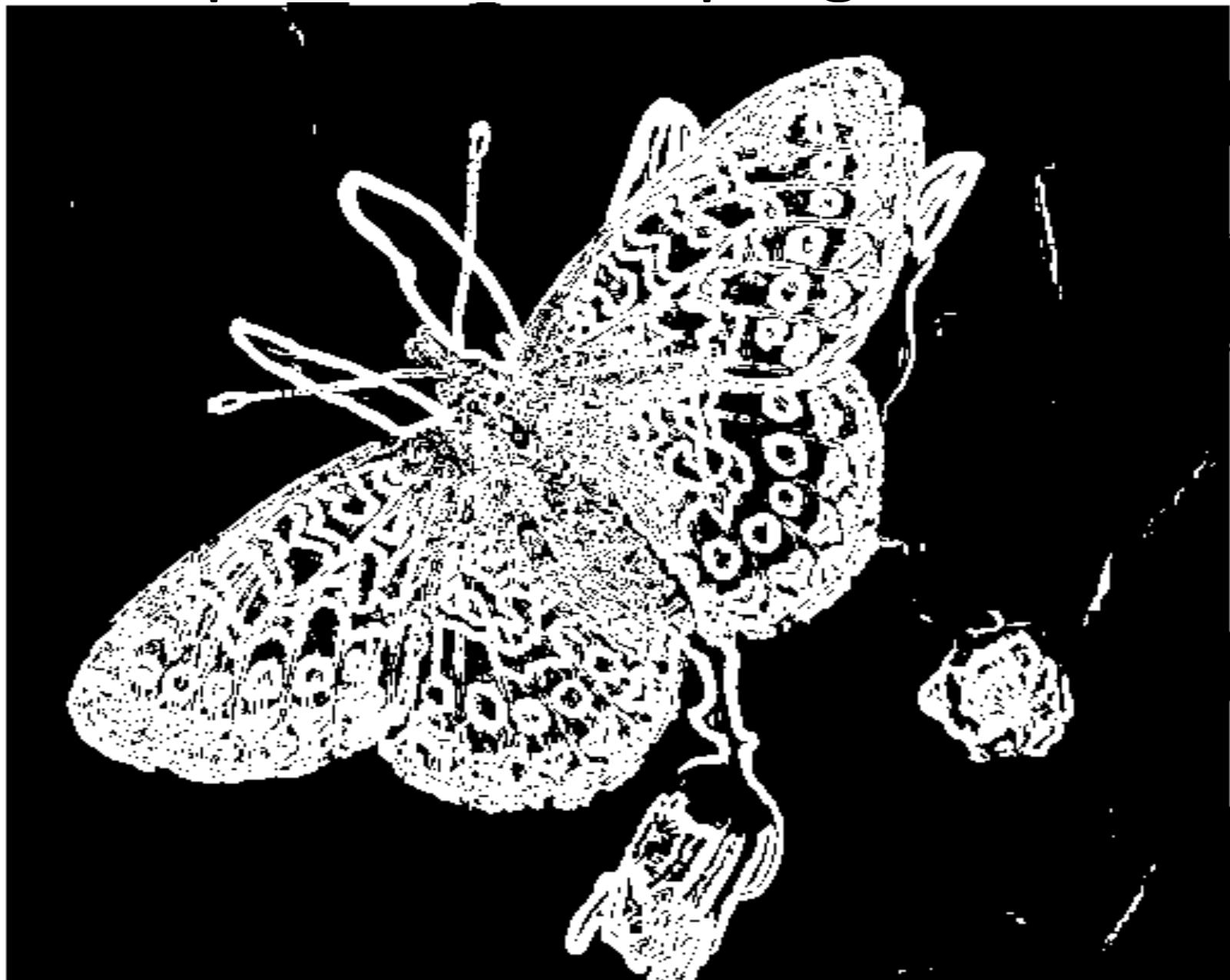
Gradientul imaginii



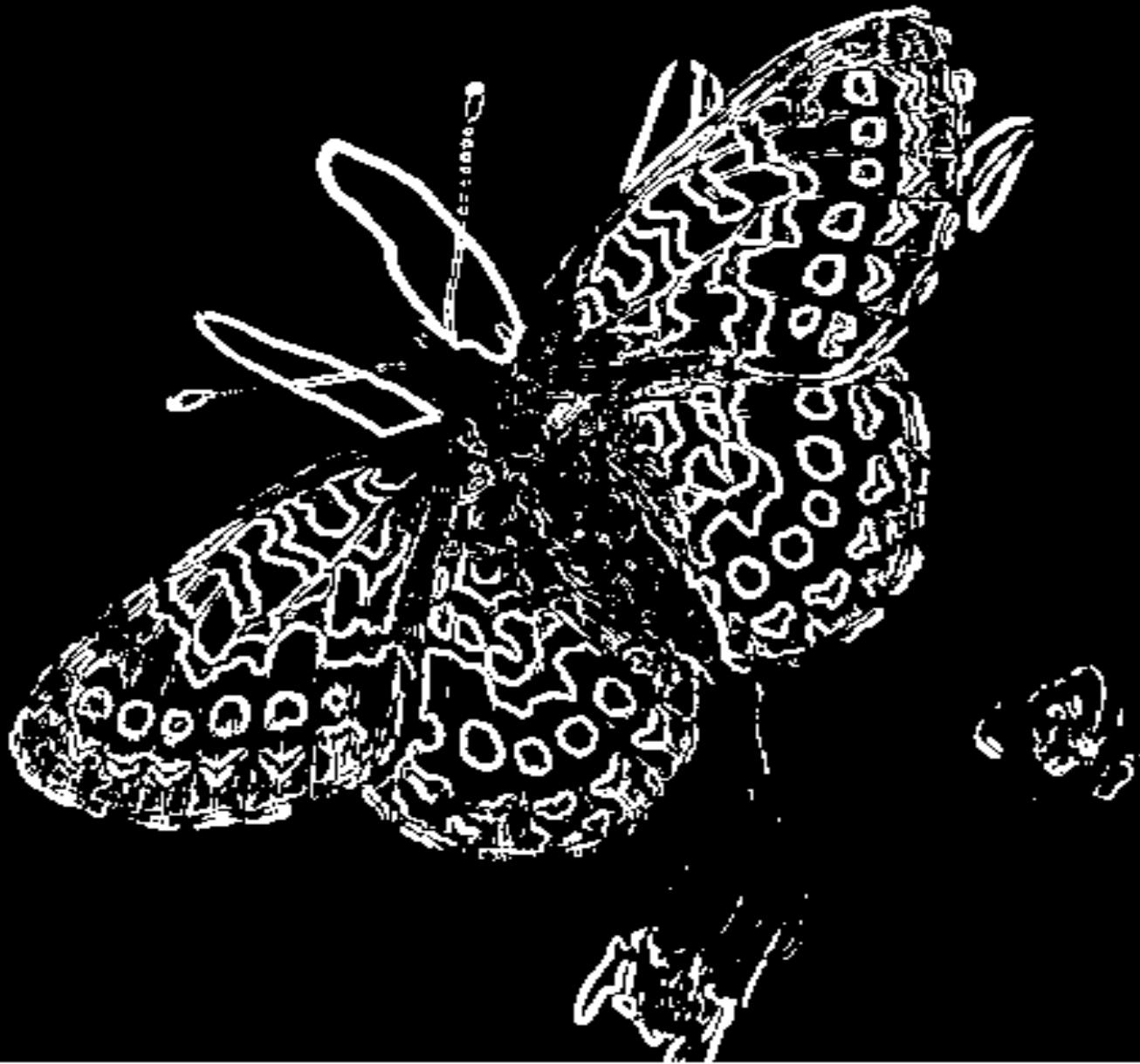
Gradientul imaginii



Aplicarea unui prag mic



Aplicarea unui prag mare



Metoda Canny

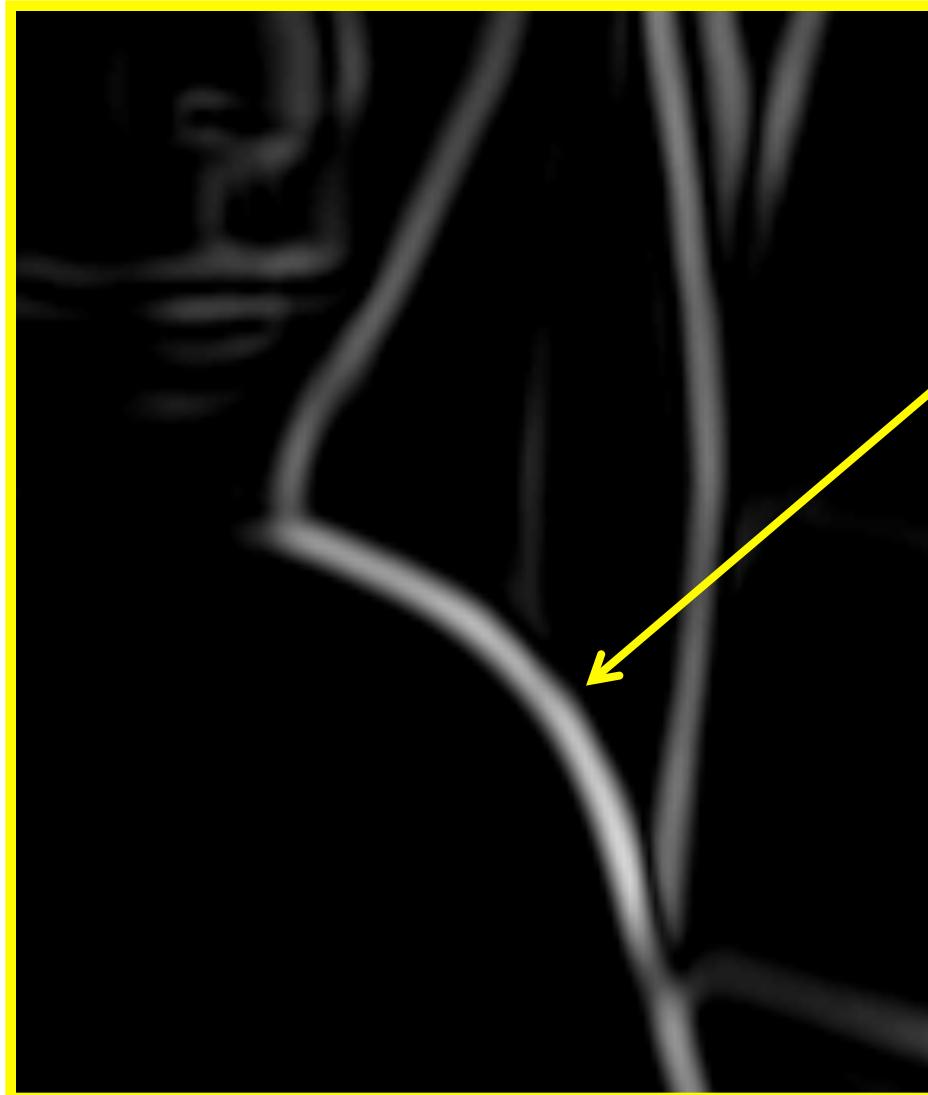


Imagine inițială



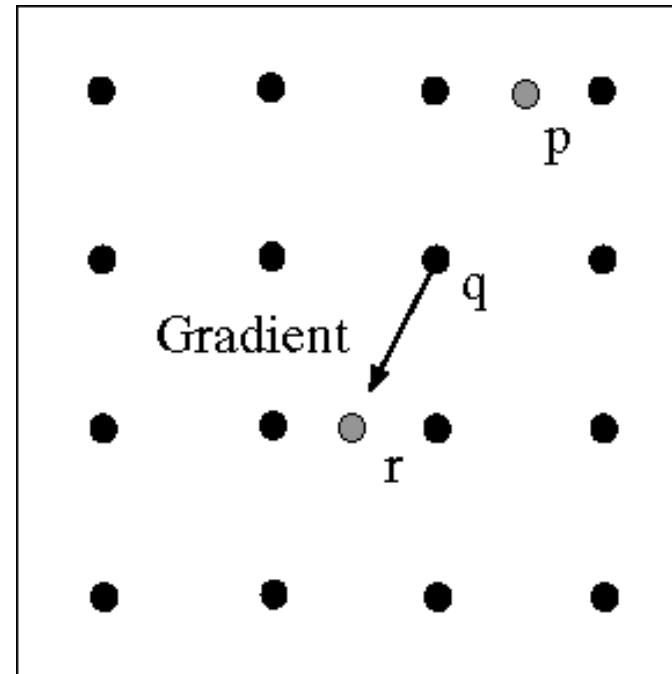
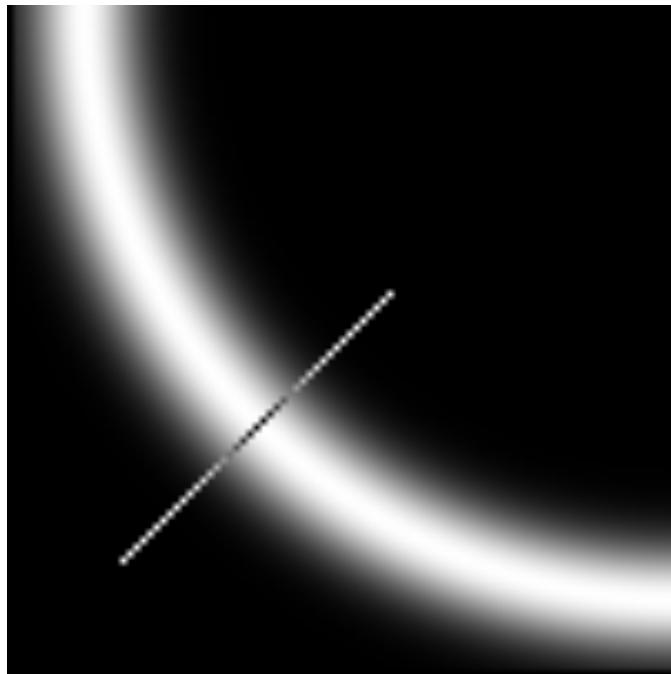
Gradientul imaginii

Metoda Canny



Cum transformăm
aceste regiuni ale
gradientului în
muchii de lățime
de un 1 pixel?

Eliminarea non-maximelor



Verifică dacă pixelul curent este maxim local de-a lungul direcției gradientului, selectează maximul de-a lungul lățimii muchiei
– necesită verificarea cu pixelii *p* și *r* obținuți prin interpolare

Metoda Canny

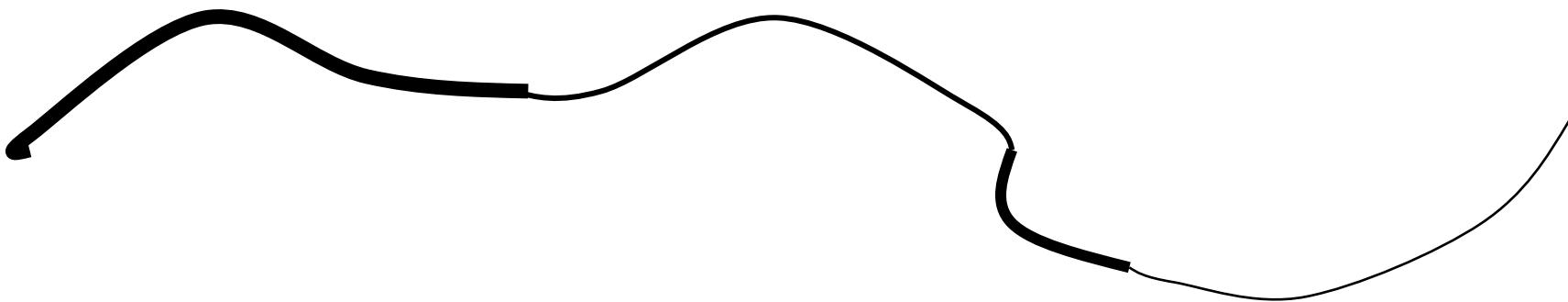


subțiere
(eliminarea non-maximelor)

Problemă:
pixelii de-a
lungul acestei
muchii nu au
“supraviețuit”

Prag mare + prag mic

- Folosește un prag de valoare mare pentru a găsi începutul conturului, apoi folosește un prag mic pentru a continua găsirea conturului.



Prag mare + prag mic



îmagine inițială



aplicarea unui prag mare
(muchii “tari”)



aplicarea unui prag mic
(muchii “slabe”)



binarizare cu histereză

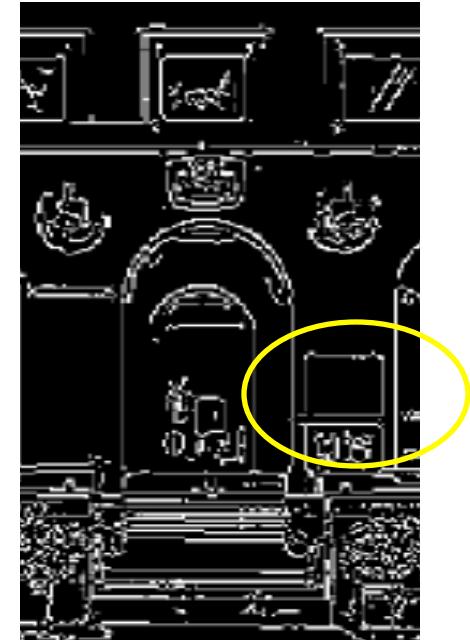
Prag mare + prag mic



aplicarea unui prag mare
(muchii "tari")



aplicarea unui prag mic
(muchii "slabe")

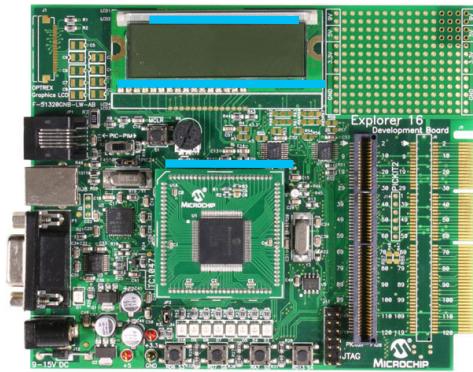


aplicarea unui prag mare
+ aplicare prag mic

Aplicație: detectarea linilor cu
transformata Hough

Aplicație: detectarea liniilor

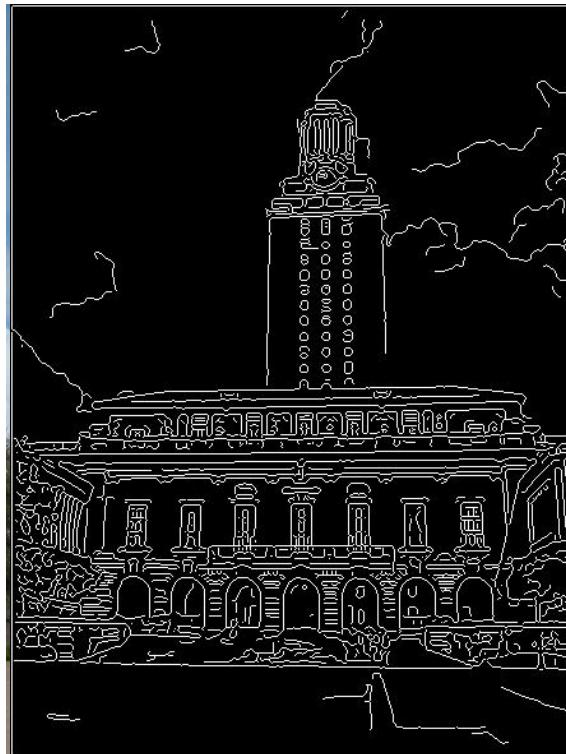
- Multe obiecte/suprafețe sunt caracterizate de prezența unor linii drepte



- De ce nu ne rezumăm la a rula un detector de muchii?

De ce e dificilă detectarea liniilor

- multe puncte din background considerate ca fiind edgels
- zgromot transformat în edgels



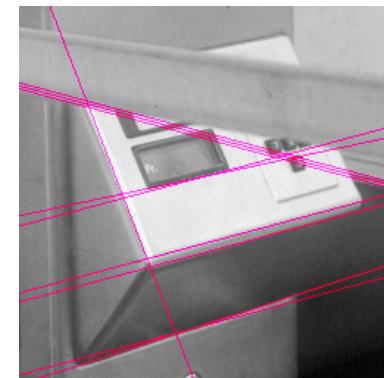
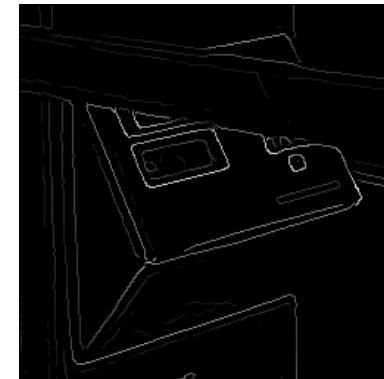
- liniile detectate parțial
- cum grupăm punctele în liniile?

Detectarea liniilor

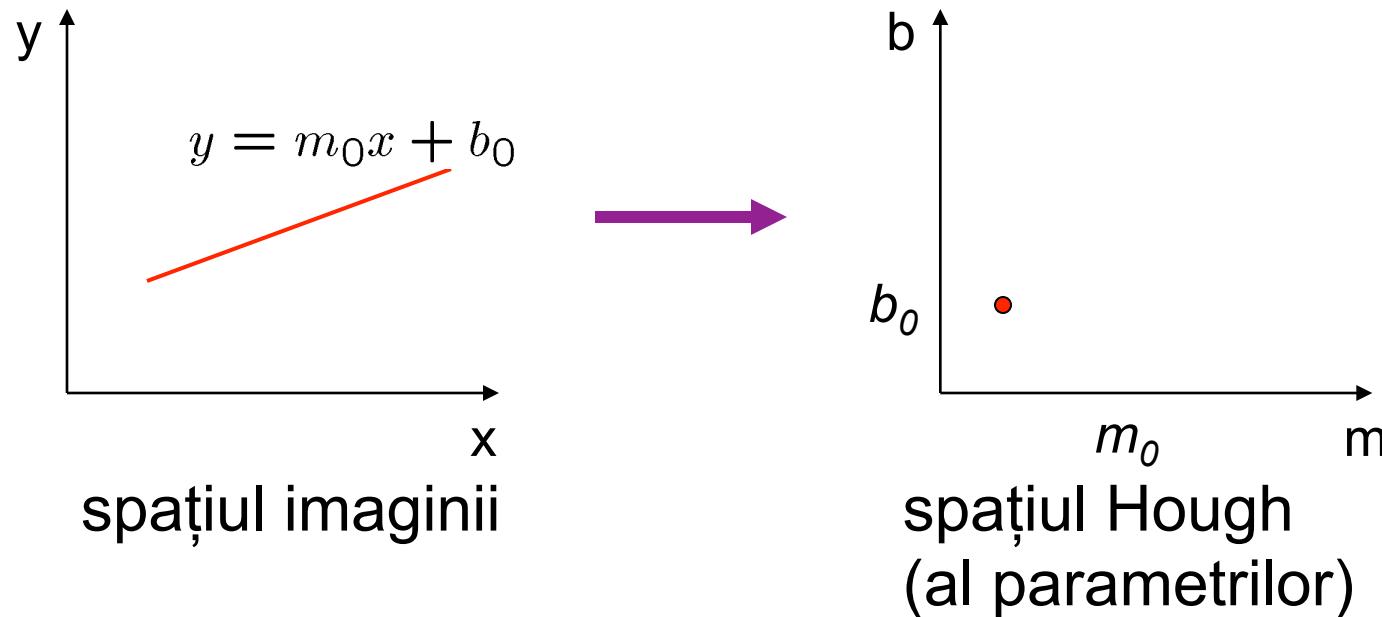
- date fiind câteva puncte care aparțin unei linii, cum găsim linia?
 - câte linii sunt în imagine?
 - fiecare puncte cărei linii aparține?
-
- **Transformata Hough** este o tehnică bazată pe “votare” care răspunde la aceste întrebări.

Ideeua principală:

1. Înregistrează toate liniile posibile care pot conține un punct detectat ca fiind edgel.
2. găsește linia care primește cele mai multe voturi.



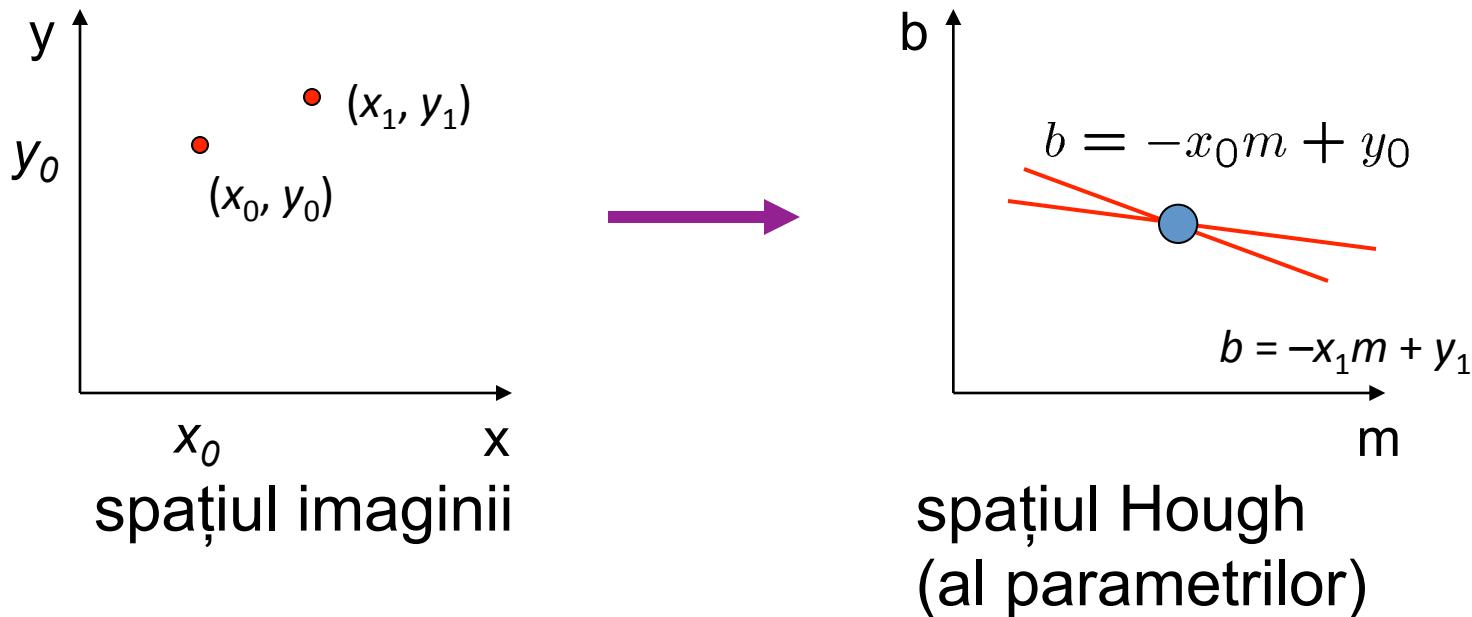
Găsirea liniilor într-o imagine: spațiul Hough



Legătura dintre spațiul imaginii (x,y) și spațiul Hough (m,b)

- o linie într-o imagine corespunde unui punct în spațiul Hough
- trecerea de la spațiul imaginii la spațiul Hough:
 - pentru un set de puncte (x,y) , găsește toate perechile (m,b) astfel încât $y = mx + b$

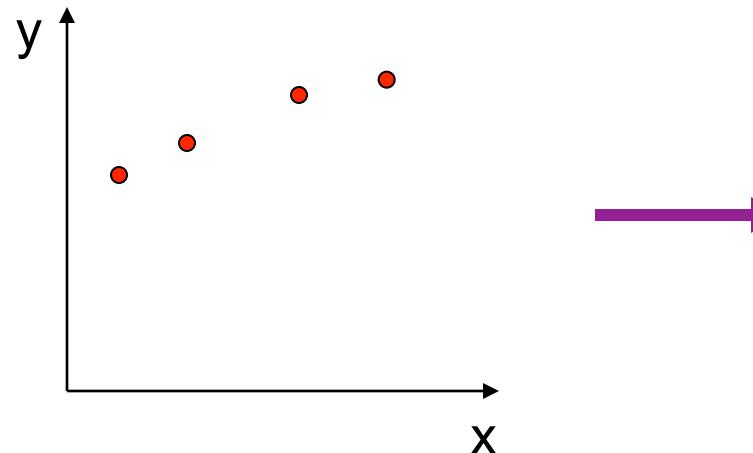
Găsirea liniilor într-o imagine: spațiul Hough



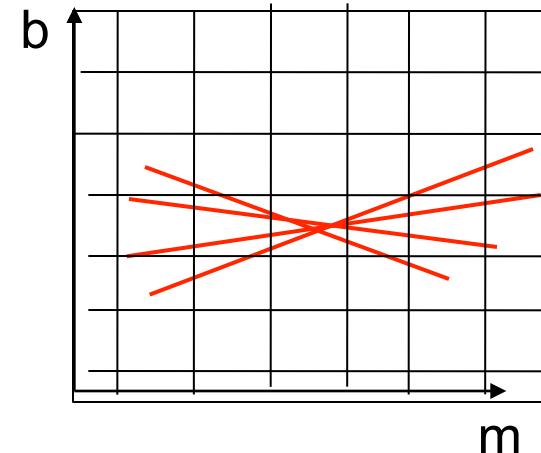
Care sunt parametrii liniei care conține punctele (x_0, y_0) și (x_1, y_1) ?

- intersecția dintre linile $b = -x_0m + y_0$ și $b = -x_1m + y_1$

Găsirea liniilor într-o imagine: algoritmul Hough



spațiu imaginii



spațiu Hough
(al parametrilor)

Cum putem folosi observația anterioară pentru a găsi parametri (m, b) ce definesc linia cea mai probabilă în spațiul imaginii?

- fiecare punct detectat ca fiind edgel în spațiu imaginii va vota pentru o mulțime de parametri în spațiu Hough
- acumulăm voturi în intervale discrete; parametri cu cel mai mare număr de voturi determină linia din spațiu imaginii