

Concepte și aplicații în Vederea Artificială

Bogdan Alexe

bogdan.alexe@fmi.unibuc.ro

Radu Ionescu

radu.ionescu@fmi.unibuc.ro

Curs opțional

anii III/IV, semestrul I, 2024-2025

Optionale an III - INFO (Lab)

Universitatea din Bucuresti, Facultatea de Matematica si Informatica, str. Academiei 14, Bucuresti

	8 8:00 - 8:50	9 9:00 - 9:50	10 10:00 - 10:50	11 11:00 - 11:50	12 12:00 - 12:50	13 13:00 - 13:50	14 14:00 - 14:50	15 15:00 - 15:50	16 16:00 - 16:50	17 17:00 - 17:50	18 18:00 - 18:50	19 19:00 - 19:50
Lu Mon			Ciocan I DezvJoc3DUnrEng5 (opt*) (Lab) Gr_1 L-303				Ciocan I DezvJoc3DUnrEng5 (opt*) (Lab) Gr_3 L-303		Paduraru C / Iordache S IntrReinfLearn Gr_1 L-309			
					Ciocan I DezvJoc3DUnrEng5 (opt*) (Lab) Gr_2 L-303		Suter F TehnSimulare (Lab) L-201		Ciocan I DezvJoc3DUnrEng5 (opt*) (Lab) Gr_4 L-303			
Ma Tue	Ciobanu A Blockchain (Lab) Gr_1 L-303								Neagu M IntrProgrJocCalc (Lab) Gr_3 L-201		Neagu M IntrProgrJocCalc (Lab) Gr_1 L-303	
			Ciobanu A Blockchain (Lab) Gr_2 L-303						Paduraru C / Iordache S IntrReinfLearn Gr_2 L-309		Tabusca S RPA UiPath (*) (Lab) Gr_1 ONLINE	
Mi Wed	Diaconu A Co&ApInVedArtif (Lab) Gr_2 L-308		Diaconu A Co&ApInVedArtif (Lab) Gr_3 L-308						industrie JavaScriptServer (*) (Lab) L-303		Neagu M IntrProgrJocCalc (Lab) Gr_2 L-201	
	Handru S / ONLINE RPA UiPath (*) (Lab) Gr_2 ONLINE								Stupariu S GraficaPeCalc (Lab) Gr_1 L-308		Dumitran M StructDateAvans (Lab) L.218	
	Paduraru C / Iordache S IntrReinfLearn Gr_3 L-303											
Jo Thu	Paduraru C / Iordache S IntrReinfLearn Gr_4 L-303		Dragan M MngAmenintCiber (Lab) L-204		Diaconu A Co&ApInVedArtif (Lab) Gr_1 L-308				Rusu C PrelucrSemnal (Lab) L-305			
	Mihailescu M SistDistrib L-308				Stupariu S GraficaPeCalc (Lab) Gr_3 L-303		Stupariu S GraficaPeCalc (Lab) Gr_2 L-308					
Vi Fri	Macovei B ImplConcLbProg (Lab) L-321		Kevorchian C TehCloudCompApiML (Lab) Gr_1 L.221A		TehCloudCompApiML (Lab) Gr_2 L.221A Kevorchian C Gr_2 ArhMasiniIntelig (Lab) Gr_2 Oana A / Marin M L-308 Blockchain (Lab) Gr_3 BanuDem. I L-303 Co&ApInVedArtif (Lab) Gr_4 Diaconu A L-322		BanuDem. I Blockchain (Lab) Gr_4 L-303		Kuvshynova O ArhMasiniIntelig (Lab) Gr_1 L-303			
							Barbu A CalcNumInfo (Lab) Gr_2 L.221A		Mihaila N CalcNumInfo (Lab) Gr_1 L.221A			

Optionale an III - INFO (Lab)

Universitatea din Bucuresti, Facultatea de Matematica si Informatica, str. Academiei 14, Bucuresti

	8 8:00 - 8:50	9 9:00 - 9:50	10 10:00 - 10:50	11 11:00 - 11:50	12 12:00 - 12:50	13 13:00 - 13:50	14 14:00 - 14:50	15 15:00 - 15:50	16 16:00 - 16:50	17 17:00 - 17:50	18 18:00 - 18:50	19 19:00 - 19:50
Lu Mon			Ciocan I DezvJoc3DUnrEng5 (opt*) (Lab) Gr_1 L-303		Ciocan I DezvJoc3DUnrEng5 (opt*) (Lab) Gr_2 L-303		Ciocan I DezvJoc3DUnrEng5 (opt*) (Lab) Gr_3 L-303		Paduraru C / Iordache S IntrReinfLearn Gr_1 L-309			
							Suter F TehnSimulare (Lab) L-201		Ciocan I DezvJoc3DUnrEng5 (opt*) (Lab) Gr_4 L-303			
Ma Tue	Ciobanu A Blockchain (Lab) Gr_1 L-303								Neagu M IntrProgrJocCalc (Lab) Gr_3 L-201		Neagu M IntrProgrJocCalc (Lab) Gr_1 L-303	
			Ciobanu A Blockchain (Lab) Gr_2 L-303						Paduraru C / Iordache S IntrReinfLearn Gr_2 L-309		Tabusca S RPA UiPath (*) (Lab) Gr_1 ONLINE	
Mi Wed	Diaconu A Co&ApInVedArtif (Lab) Gr_2 L-308		Diaconu A Co&ApInVedArtif (Lab) Gr_3 L-308						industrie JavaScriptServer (*) (Lab) L-303		Neagu M IntrProgrJocCalc (Lab) Gr_2 L-201	
	Handru S / ONLINE RPA UiPath (*) (Lab) Gr_2 ONLINE								Stupariu S GraficaPeCalc (Lab) Gr_1 L-308		Dumitran M StructDateAvans (Lab) L.218	
	Paduraru C / Iordache S IntrReinfLearn Gr_3 L-303											
Jo Thu	Paduraru C / Iordache S IntrReinfLearn Gr_4 L-303		Dragan M MngAmenintCiber (Lab) L-204		Diaconu A Co&ApInVedArtif (Lab) Gr_1 L-308				Rusu C PrelucrSemnal (Lab) L-305			
	Mihailescu M SistDistrib L-308				Stupariu S GraficaPeCalc (Lab) Gr_3 L-303		Stupariu S GraficaPeCalc (Lab) Gr_2 L-308					
Vi Fri	Macovei B ImplConcLbProg (Lab) L-321		Kevorchian C TehCloudCompApIML (Lab) Gr_1 L.221A		TehCloudCompApIML (Lab) Gr_2 L.221A		BanuDem. I Blockchain (Lab) Gr_4 L-303		Kuvshynova O ArhMasiniIntelig (Lab) Gr_1 L-303			
					ArhMasiniIntelig (Lab) Gr_2 L-308							
					Oana A / Marin M Blockchain (Lab) Gr_3 L-308		Barbu A CalcNumInfo (Lab) Gr_2 L.221A		Mihaila N CalcNumInfo (Lab) Gr_1 L.221A			
					BanuDem. I Co&ApInVedArtif (Lab) Gr_4 L-303							
					Diaconu A L-322							

Orar generat:13.10.2024

aSc Orare

Cursul trecut

- Diverse modele pentru zgomot în imagini
 - salt and pepper, impuls
 - Gaussian (normal)
- Filtrarea liniară
 - corelație, convoluție
 - filtre: de medie, Gaussian, accentuare
 - aplicație: imagini hibrid
- Filtrarea neliniară
 - filtrul median



normal

Cursul de azi

- Aplicații ale filtrelor:

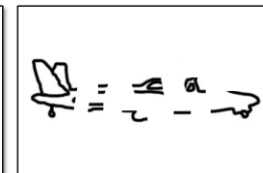
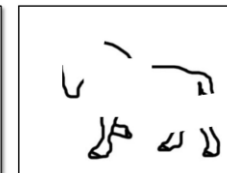
- găsirea șabloanelor (cursul de azi)



- redimensionarea imaginilor (cursul de azi)



- extragerea informației (muchii, textură – cursul de azi + săptămâna viitoare)



Găsirea șabloanelor (template matching)

Găsirea șabloanelor - exemplu



Vrem să găsim în imaginea alăturată șablonul de mai jos:



șablon
(sens unic)

- cea mai simplă metodă de detectare a obiectelor
- pentru fiecare pixel din imagine, compară șablonul cu fereastra centrată la acel pixel: măsoară cât de bine se aseamănă fereastra cu șablonul (similaritate)

Găsirea șabloanelor - exemplu



Vrem să găsim în imaginea alăturată șablonul de mai jos:



șablon
(sens unic)

- cea mai simplă metodă de detectare a obiectelor
- pentru fiecare pixel din imagine, compară șablonul cu fereastra centrată la acel pixel: măsoară cât de bine se aseamănă fereastra cu șablonul (similaritate)

Găsirea șabloanelor - exemplu

Similaritate



Valoarea cea mai mare a similarității



șablon
(sens unic)

- cea mai simplă metodă de detectare a obiectelor
- pentru fiecare pixel din imagine, compară șablonul cu fereastra centrată la acel pixel: măsoară cât de bine se aseamănă fereastra cu șablonul (similaritate)

Găsirea șabloanelor - exemplu

Similaritate



Valoarea cea mai mare a similarității



șablon
(sens unic)

- similaritate = imagine filtrată (șablon = filtru)
- o măsură bună pentru similaritatea a două ferestre?
 - corelația?
 - suma pătratelor distanțelor?
 - altceva?

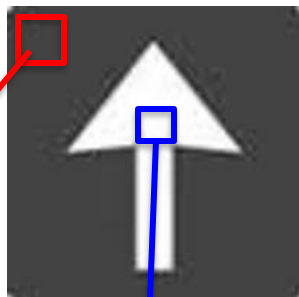
Găsirea șabloanelor cu filtre

- șablon = 

rgb2gray
cv.cvtColor



67 67 67 67 67
67 67 67 67 67
67 67 67 67 67
67 67 67 67 67



254 254 254 254
254 254 254 254
254 254 254 254
254 254 254 254

rgb2gray = **cv.cvtColor**



Găsirea șabloanelor cu filtre

- filtrul $f_1 = \uparrow$
- corelație: filtrăm imaginea cu filtrul f_1

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l f_1(u, v) I(i + u, j + v)$$



I

$$O_1 = f_1 \otimes I$$

Ce rezultă după filtrare?

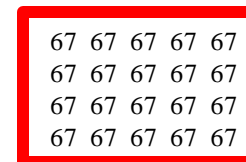
Găsirea șabloanelor cu filtre

- filtrul $f_1 = \begin{bmatrix} \uparrow \end{bmatrix}$
- corelație: filtrăm imaginea cu filtrul f_1

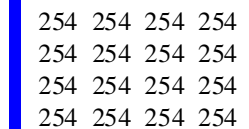
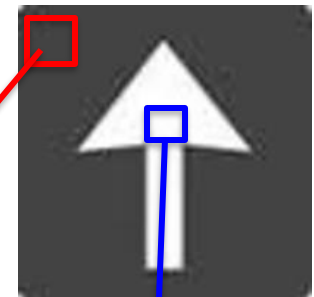
$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l f_1(u, v) I(i + u, j + v)$$

Explicații?

Filtru ne-normalizat
(suma > 1) rezultă în
imagine filtrată cu
luminozitate crescută



67	67	67	67	67
67	67	67	67	67
67	67	67	67	67
67	67	67	67	67
67	67	67	67	67



254	254	254	254	254
254	254	254	254	254
254	254	254	254	254
254	254	254	254	254
254	254	254	254	254

imagine complet albă

O_1

Găsirea șabloanelor cu filtre

- filtrul $f_2 = \begin{bmatrix} \uparrow \end{bmatrix}$ normalizat (suma = 1: 67 \rightarrow 0.00007, 254 \rightarrow 0.0003)
- corelație: filtrăm imaginea cu filtrul f_2

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l f_2(u, v) I(i + u, j + v)$$



I

$$O_2 = f_2 \otimes I$$

Ce rezultă după filtrare?

Găsirea șabloanelor cu filtre

- filtrul $f_2 = \begin{bmatrix} \uparrow \end{bmatrix}$ normalizat (suma = 1: 67 \rightarrow 0.00007, 254 \rightarrow 0.0003)
- corelație: filtrăm imaginea cu filtrul f_2

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l f_2(u, v) I(i + u, j + v)$$



Explicații?

Răspunsul filtrului este mare pentru intensități mari în imagine. Dacă am avea o porțiune din imagine numai cu alb acolo am obține cea mai mare valoare (răspunsul filtrului) pentru pixelii din O_2

O_2

Găsirea șabloanelor cu filtre

- filtrul $f_3 = \begin{bmatrix} \uparrow \end{bmatrix}$ normalizat (suma = 0, $f_3 = f_2 - \overline{f_2}$)
- corelație: filtrăm imaginea cu filtrul f_3

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l f_3(u, v) I(i + u, j + v)$$



I

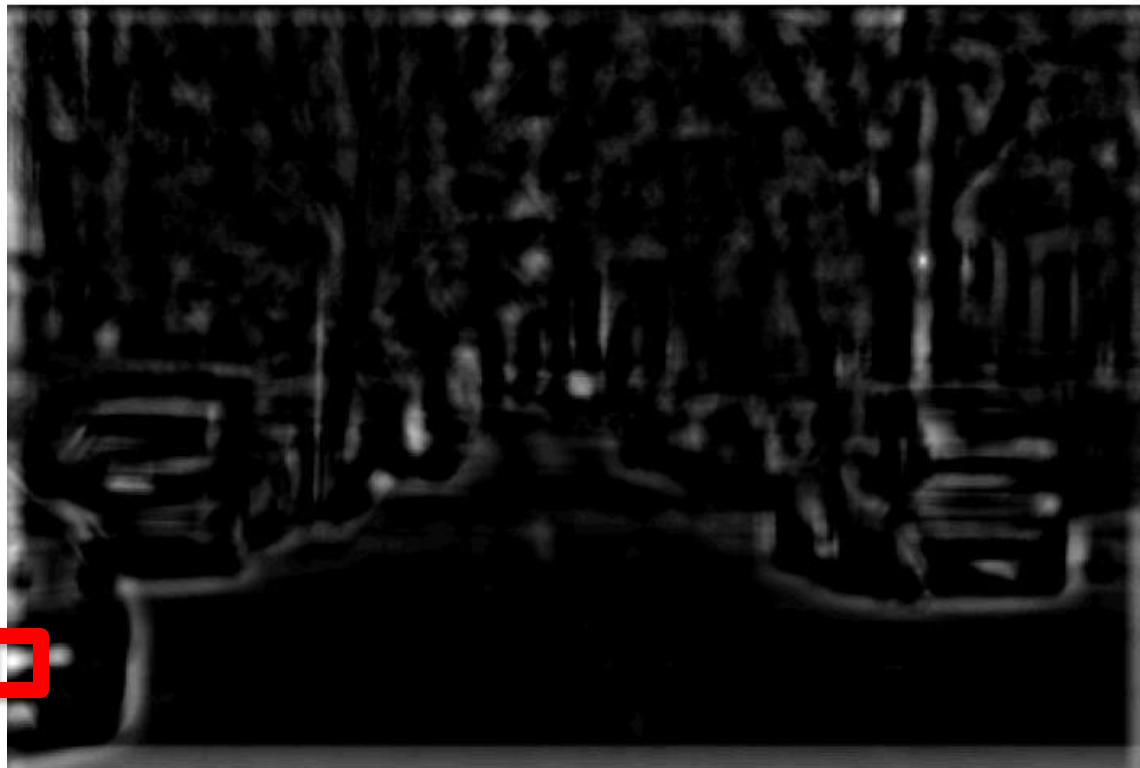
$$O_3 = f_3 \otimes I$$

Ce rezultă după filtrare?

Găsirea șabloanelor cu filtre

- filtrul $f_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ normalizat (suma = 0, $f_3 = f_2 - \overline{f_2}$)
- corelație: filtrăm imaginea cu filtrul f_3

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l f_3(u, v) I(i + u, j + v)$$



Explicații?

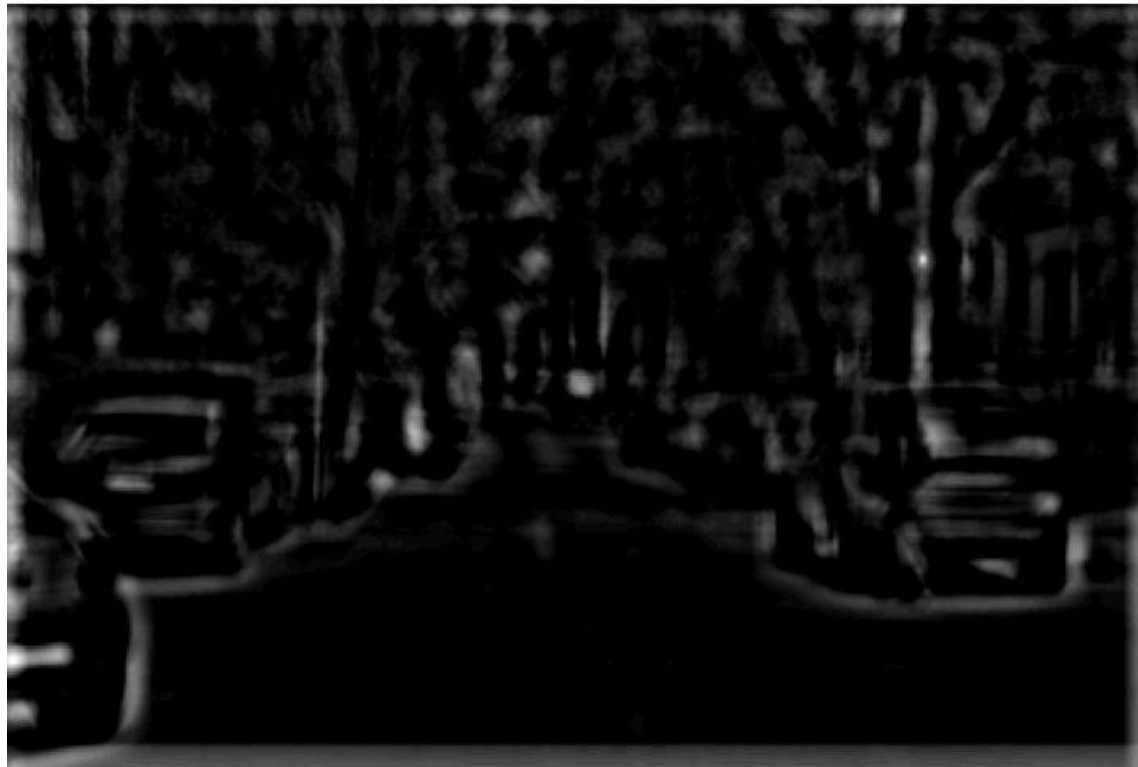
Răspunsul filtrului este mare atunci când valorilor mici (negative) din filtru corespund intensități mici (pixeli negri) din imagine, iar valorilor mari (pozitive) din filtru corespund intensități mari (pixeli albi) din imagine.

O_3

Găsirea șabloanelor cu filtre

- filtrul $f_3 = \begin{bmatrix} \uparrow \end{bmatrix}$ normalizat (suma = 0, $f_3 = f_2 - \overline{f_2}$)
- corelație: filtrăm imaginea cu filtrul f_3

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l f_3(u, v) I(i + u, j + v)$$



Explicații?

Răspunsul filtrului este maxim atunci când avem în imagine o fereastră de dimensiunile filtrului cu pixeli albi și negri corelați cu semnele +/- ale valorilor filtrului.

O_3

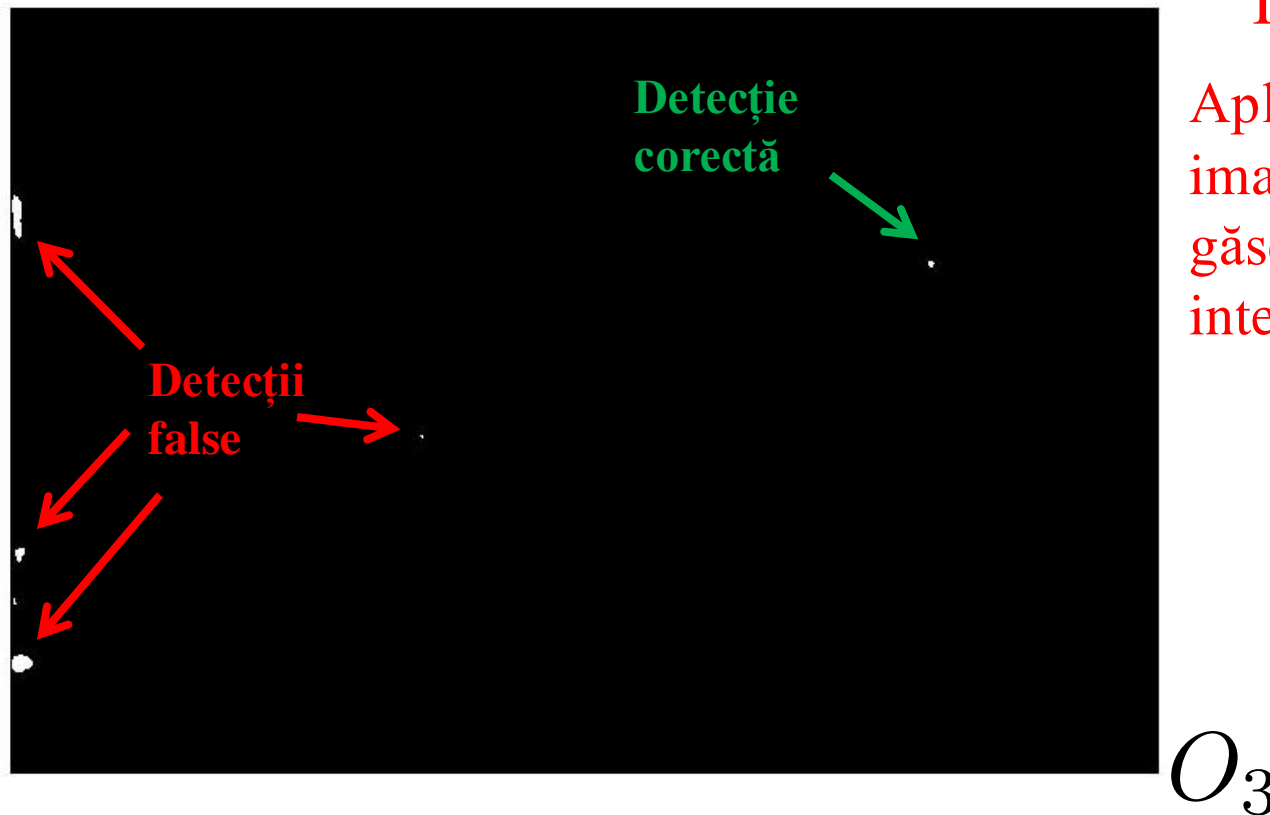
Găsirea șabloanelor cu filtre

- filtrul $f_3 = \begin{bmatrix} 1 \\ \uparrow \end{bmatrix}$ normalizat (suma = 0, $f_3 = f_2 - \overline{f_2}$)
- corelație: filtrăm imaginea cu filtrul f_3

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l f_3(u, v) I(i + u, j + v)$$

Thresholding

Aplicăm un prag (threshold) imaginii filtrate obținute:
găsește toți pixelii cu
intensitate > threshold



Găsirea șabloanelor cu filtre

- filtrul $f_1 = \uparrow$
- suma pătratelor distanțelor

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - f_1(u, v))^2$$



I

Găsirea șabloanelor cu filtre

- filtrul $f_1 = \uparrow$
- suma pătratelor distanțelor

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - f_1(u, v))^2$$



Găsește valoarea minimă a distanței/maximă a similarității detectând șablonul.

O_4 (1 – distanta)

Găsirea șabloanelor cu filtre

- filtrul $f_1 = \uparrow$
- suma pătratelor distanțelor

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - f_1(u, v))^2$$



Efect de umbrire al imaginii
care afectează șablonul.

Ce obținem?

Găsirea șabloanelor cu filtre

- filtrul $f_1 = \uparrow$
- suma pătratelor distanțelor

$$o(i, j) = \sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - f_1(u, v))^2$$



Explicații?

Răspunsul filtrului este
senzitiv la intensitatea
medie din fereastră

O_5 (1 – distanta)

Găsirea șabloanelor cu filtre

- filtrul $f_1 = \uparrow$

`res = cv.matchTemplate(img,template,method)`

- corelație normalizată (de medie 0)

$$o(i, j) = \frac{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})(f_1(u, v) - \bar{f}_1)}{\sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})^2} \sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (f_1(u, v) - \bar{f}_1)^2}}$$

media intensităților în fereastra curentă din imagine

media intensităților filtrului

deviația standard a intensităților în fereastra curentă din imagine

deviația standard a intensităților în filtru

Cosinusul a doi vectori normalizați

Invarianță la transformări fotometrice

- filtrul $f_1 = \uparrow$

`res = cv.matchTemplate(img,template,method)`

- corelație normalizată

$$o(i, j) = \frac{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})(f_1(u, v) - \bar{f}_1)}{\sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})^2} \sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (f_1(u, v) - \bar{f}_1)^2}}$$

- modificarea luminozității (brightness \rightarrow b):

$$I_{[u-k:u+k] \times [v-l:v+l]} = f_1 + b$$

- modificarea contrastului (contrast \rightarrow c): :

$$I_{[u-k:u+k] \times [v-l:v+l]} = c \times f_1$$

- modificare luminozitate + contrast:

$$I_{[u-k:u+k] \times [v-l:v+l]} = b + c \times f_1$$

- în toate aceste cazuri $o(i, j)$ ia aceeași valoare = 1

Găsirea șabloanelor cu filtre

- filtrul $f_1 = \uparrow$

`res = cv.matchTemplate(img,template,method)`

- corelație normalizată

$$o(i, j) = \frac{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})(f_1(u, v) - \bar{f}_1)}{\sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})^2} \sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (f_1(u, v) - \bar{f}_1)^2}}$$



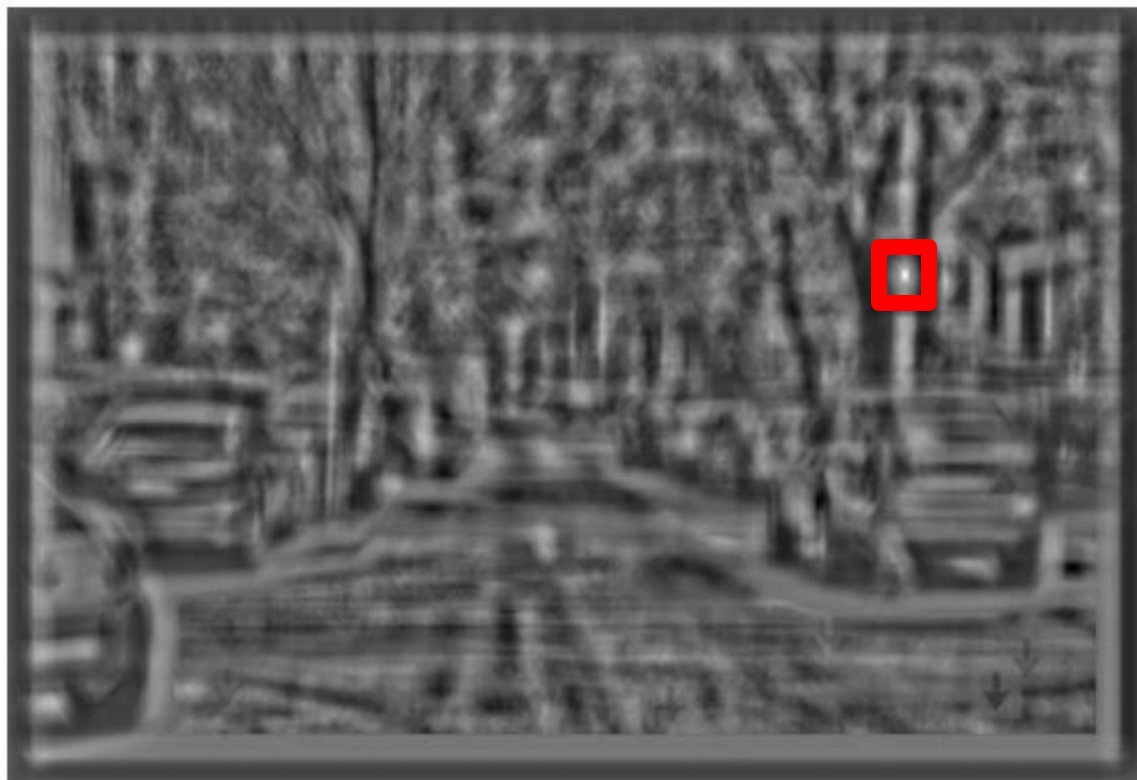
Găsirea șabloanelor cu filtre

- filtru $f_1 = \uparrow$

`res = cv.matchTemplate(img,template,method)`

- corelație normalizată

$$o(i, j) = \frac{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})(f_1(u, v) - \bar{f}_1)}{\sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})^2} \sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (f_1(u, v) - \bar{f}_1)^2}}$$



Găsește valoarea minimă
detectând șablonul.

Găsirea șabloanelor cu filtre

- filtru $f_1 = \uparrow$

`res = cv.matchTemplate(img,template,method)`

- corelație normalizată

$$o(i, j) = \frac{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})(f_1(u, v) - \bar{f}_1)}{\sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})^2} \sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (f_1(u, v) - \bar{f}_1)^2}}$$



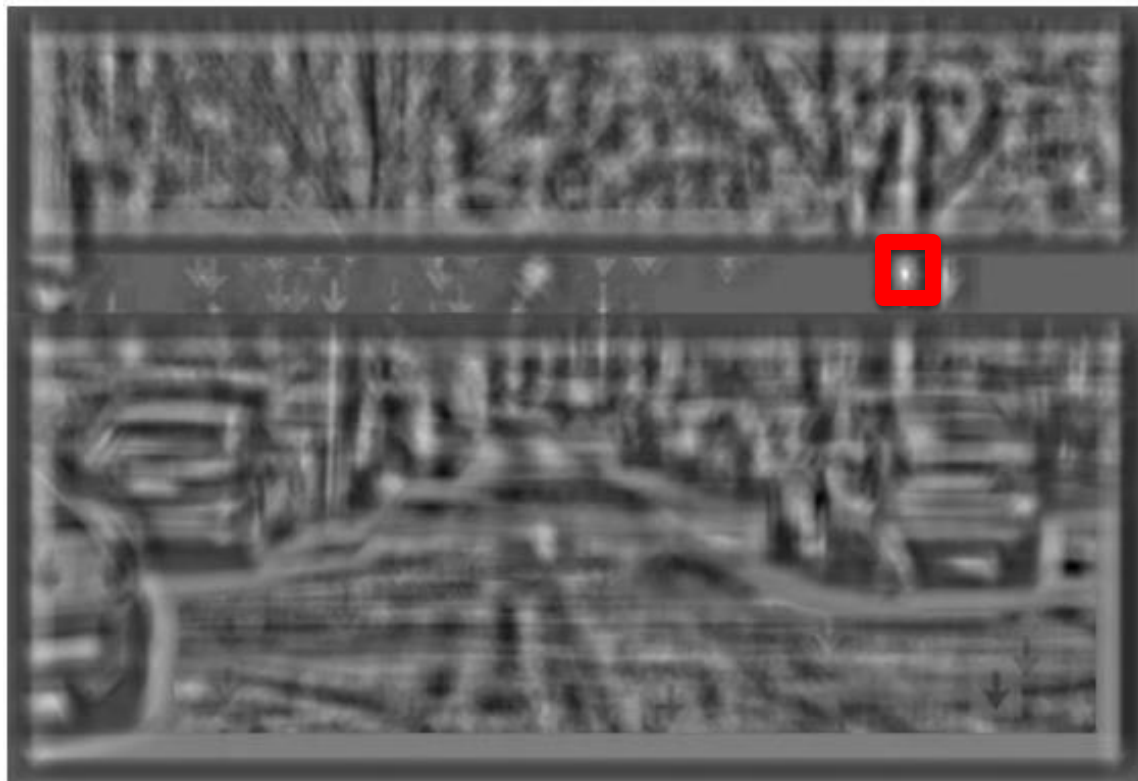
Găsirea șabloanelor cu filtre

- filtru $f_1 = \uparrow$

`res = cv.matchTemplate(img,template,method)`

- corelație normalizată

$$o(i, j) = \frac{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})(f_1(u, v) - \bar{f}_1)}{\sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (I(i+u, j+v) - \bar{I}_{u,v})^2} \sqrt{\sum_{u=-k}^k \sum_{v=-l}^l (f_1(u, v) - \bar{f}_1)^2}}$$



Găsește valoarea minimă
detectând șablonul.

Waldo



Where's Waldo?

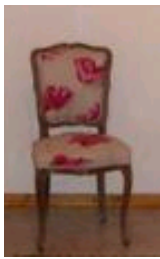
Waldo



Cea mai bună măsură a similarității?

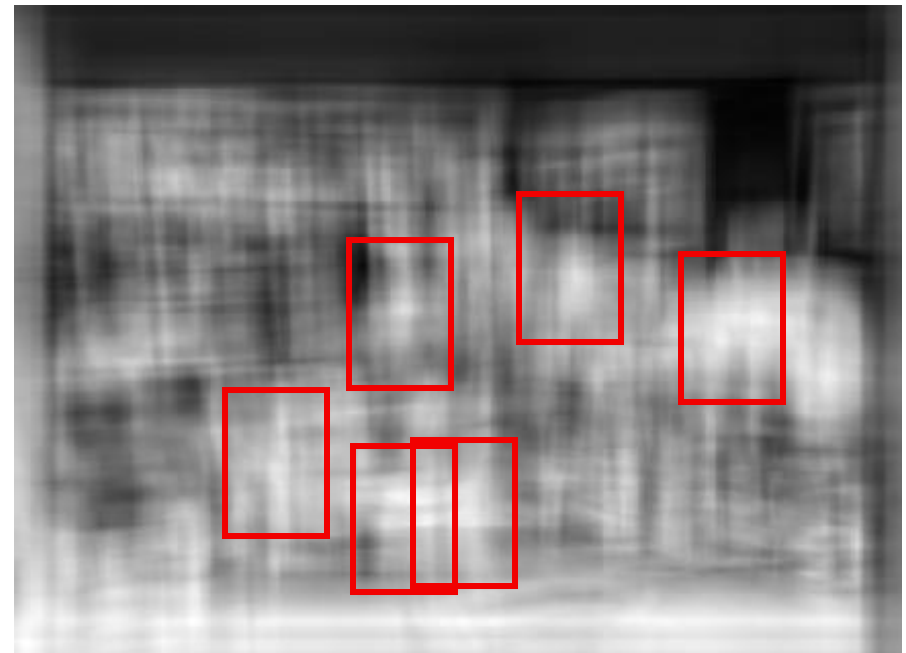
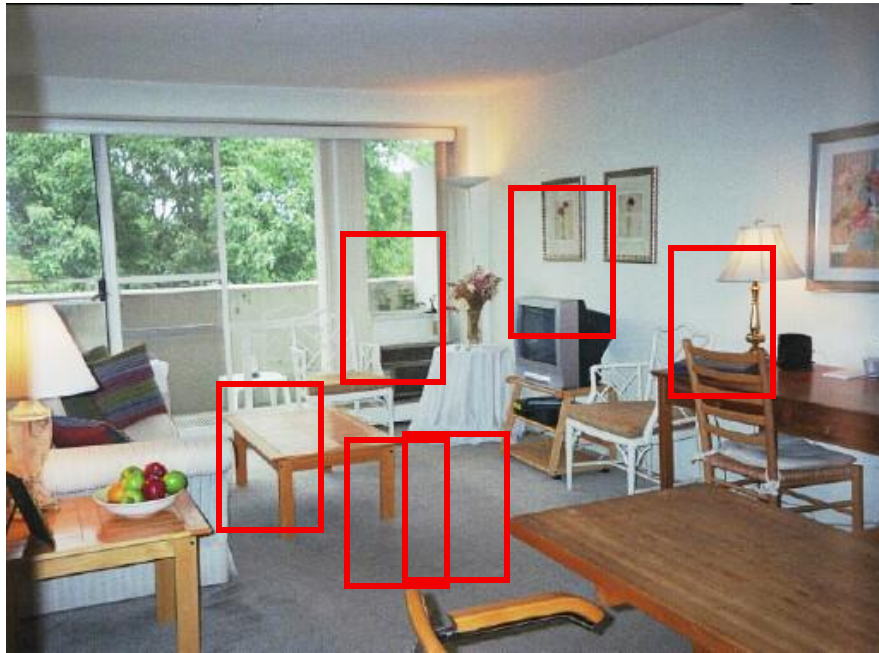
- filtru de medie 0: rezultate nu prea bune (apar detecții false)
- suma pătratelor distanțelor: sensibil la intensitatea medie
- corelație normalizată: invariant la intensitatea medie, luminozitate și la contrast

viteză de execuție



Detectare de obiecte cu găsierea șabloanelor

Găsiți scaunele în această imagine



Găsirea șabloanelor nu rezolvă problema

A “popular method is that of template matching, by point to point correlation of a model pattern with the image pattern. These techniques are inadequate for three-dimensional scene analysis for many reasons, such as occlusion, changes in viewing angle, and articulation of parts.” Nevatia & Binford, 1977.

Detectare de obiecte cu găsirea șabloanelor


Avantaje

- metodă simplă, ușor de implementat
- rezultate bune când șablonul/ceva foarte asemănător cu șablonul se află în imagine

Dezavantaje

- nu este invariant la mărime, rotație (chiar a aceluiași șablon): dacă mărim șablonul sau îl rotim nu mai obținem același rezultat
- pentru clase de obiecte cu variabilitate în înfățișare foarte mare (mașini, persoane) metoda nu e aplicabilă
- nu putem să folosim această metodă la detectarea facială

Dacă vrem să găsim un șablon mai mic sau mai mare?

șablon inițial 

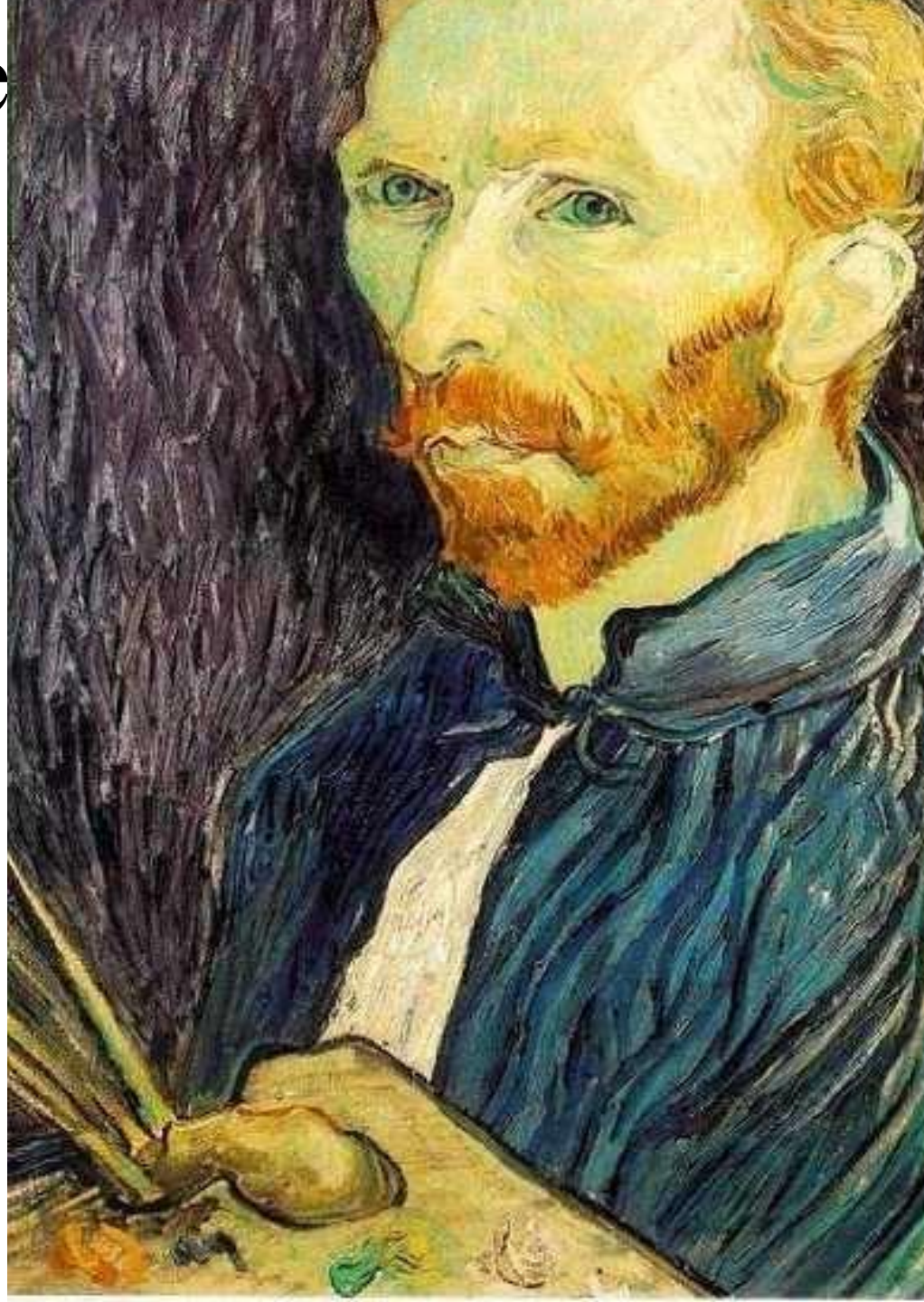
șablon mai mic   alternativă construim o piramidă de imagini redimensionate



Redimensionarea imaginilor

Micșorare

Această imagine este prea mare pentru a încăpea pe slide. Cum putem genera o imagine de 2× mai mică?

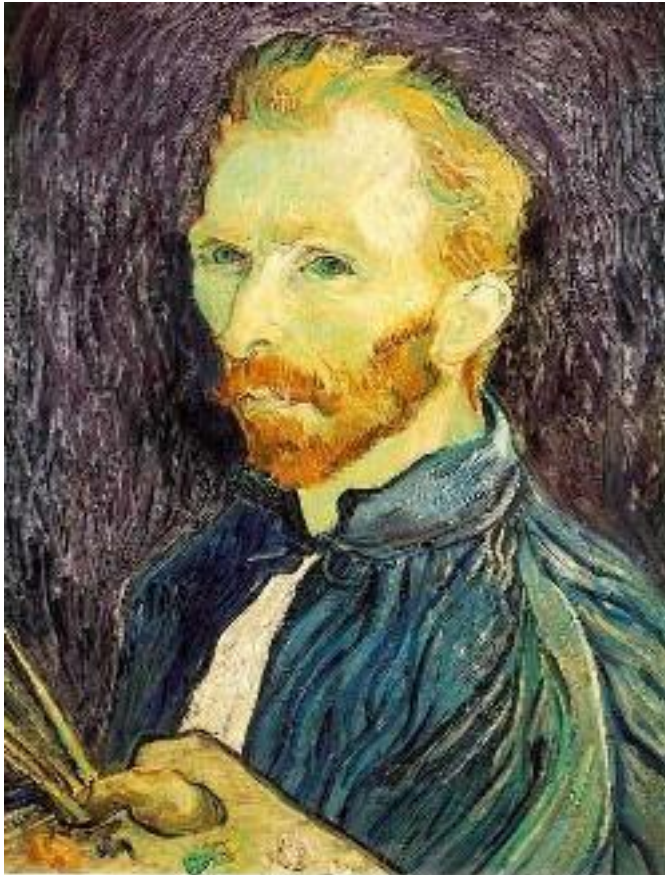


Algoritm pentru micșorarea imaginilor cu factor 2

1. Input: imagine de dimensiuni $L \times C$
2. Selectează fiecare al doilea pixel

`imgRedusa = img [0::2, 0::2]`

Rezultate



1/4



1/8

Eliminăm fiecare a doua linie și a doua coloană pentru a crea o imagine de 2× mai mică

Rezultate - zoom



1/2



1/4 (2× zoom)



1/8 (4× zoom)

Observații?

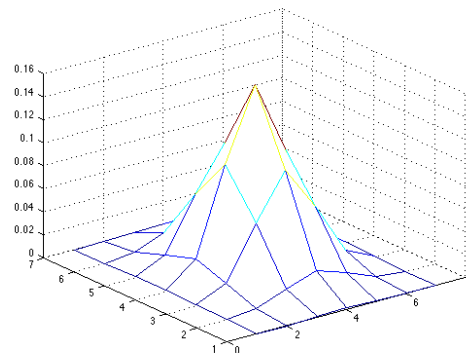
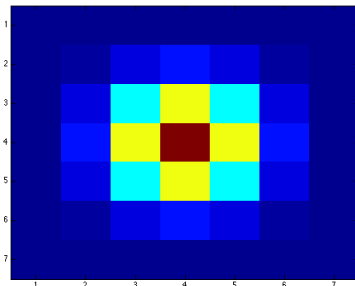
Algoritm pentru micșorarea imaginilor cu factor 2 cu filtrare Gaussiană

1. Input: imagine de dimensiuni $L \times C$
2. Filtrează imaginea cu un filtru Gaussian

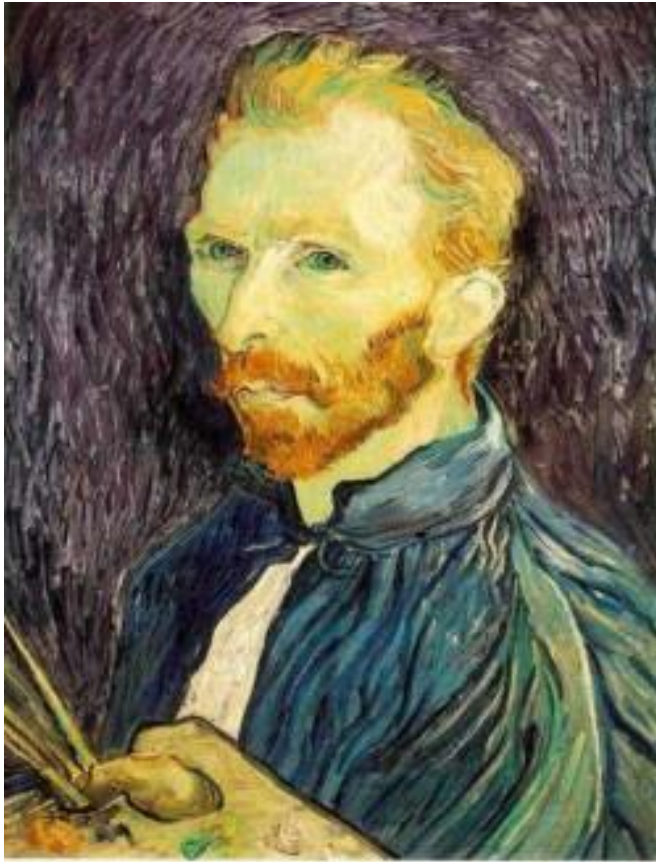
`imgBlurata = cv2.GaussianBlur(img,(5,5),0)`

3. Selectează fiecare al doilea pixel

`imgRedusa = imgBlurata [0::2, 0::2]`



Rezultate



Gaussian $1/2$



G $1/4$



G $1/8$

Rezultate - zoom



Gaussian $1/2$

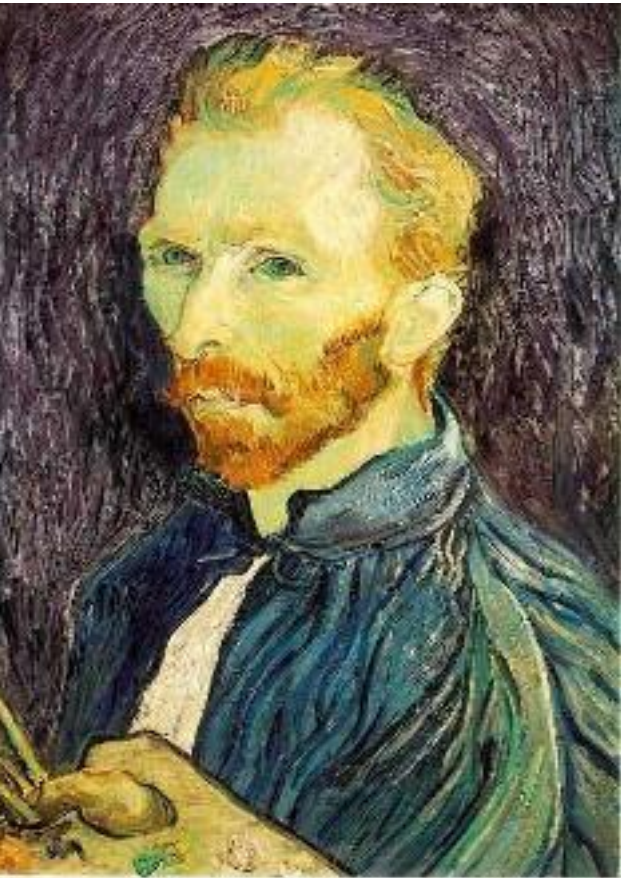


G $1/4$



G $1/8$

Versus...



$1/2$



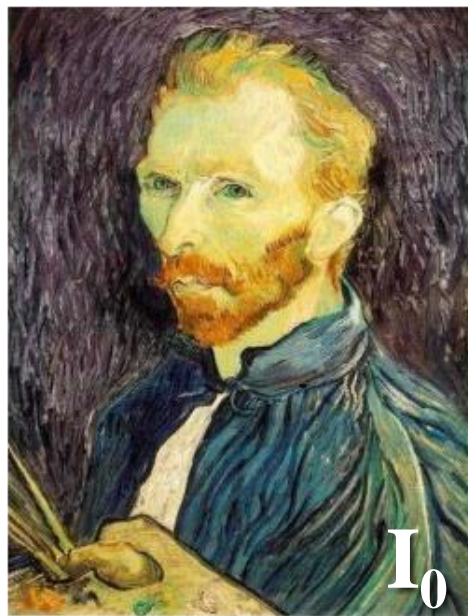
$1/4$ (2× zoom)



$1/8$ (4× zoom)

Filtrare Gaussiană

- Soluție: filtrează
mai întâi imaginea
cu un filtru
Gaussian, *apoi*
selectează pixelii
din 2 în 2



blurează

selectează

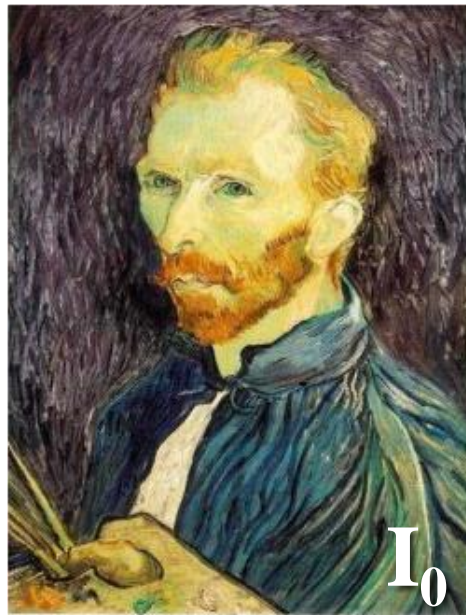
blurează

selectează

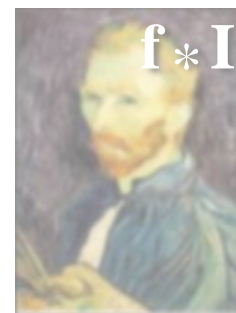
• • •



*Piramidă
Gaussiană*



• • •

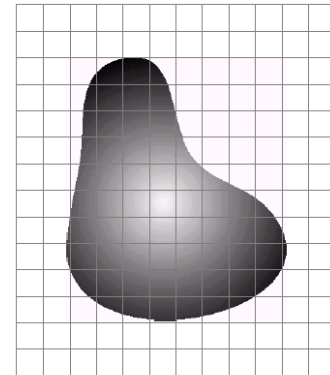
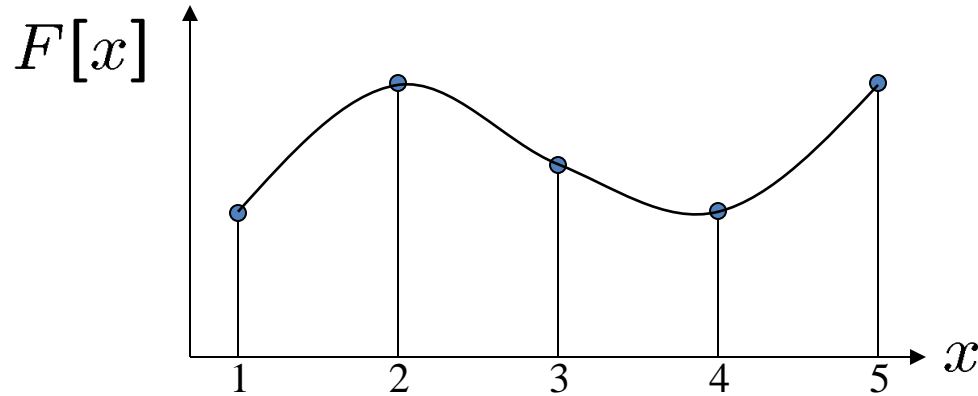


Mărirea imaginilor

- imagine prea mică pentru ecran: (45 x 45 pixeli)
- cum putem să o mărim de 10 ori? (450 x 450 pixeli)
- metodă simplă: repetăm fiecare rând și coloană de 10 ori (fiecare pixel multiplicat de 100 de ori)
- interpolare “cel mai apropiat vecin”

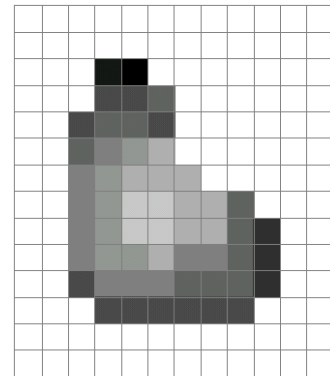


Interpolarea imaginilor



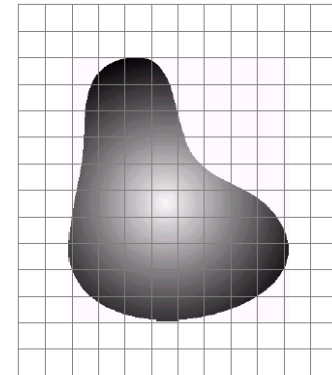
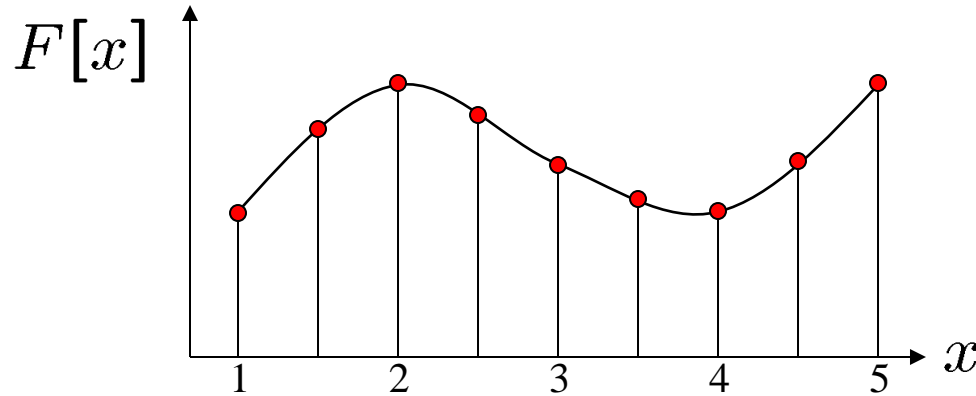
f - funcție continuă

- Cum se formează imaginile digitale?
- eșantionare : discretizăm spațiul în pixeli
 - trecem de la o **funcție continuă f** la o **funcție discretă F**
 - dacă am putea reconstrui funcția continuă inițială f , am putea genera imagini la orice rezoluție



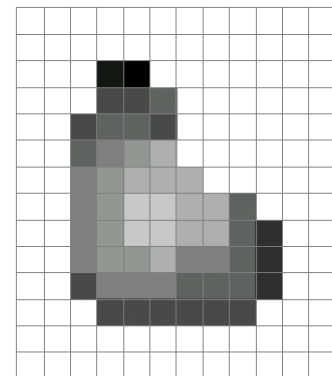
F - funcție discretă

Interpolarea imaginilor



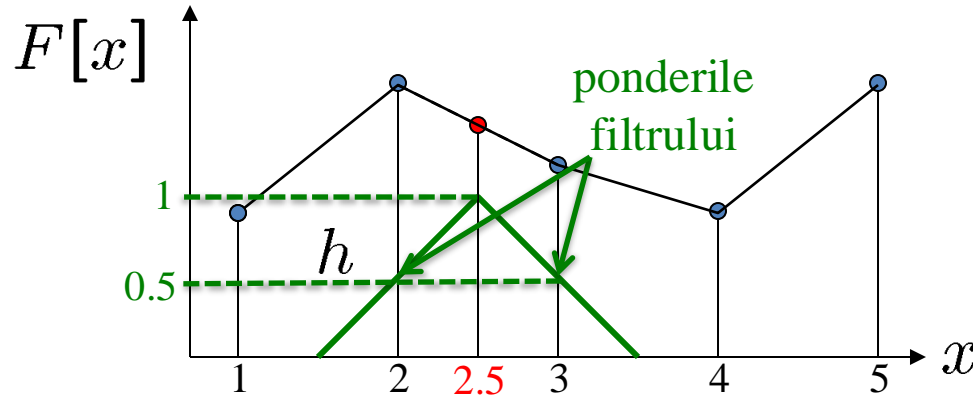
f - funcție continuă

- Cum se formează imaginile digitale?
- eșantionare : discretizăm spațiul în pixeli
 - trecem de la o funcție continuă f la o funcție discretă F
 - dacă am putea reconstrui funcția continuă inițială f , am putea genera imagini la orice rezoluție



F - funcție discretă

Interpolarea imaginilor



h – filtru liniar

- Dacă nu cunoaștem f ?

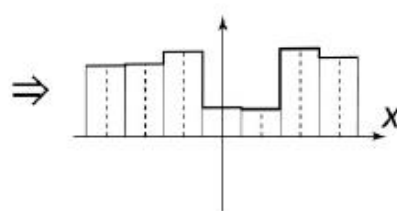
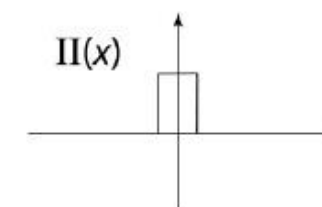
- aproximăm f : \tilde{f}
- folosim filtrarea

- convertim F în funcția $f_F = \begin{cases} F(x), & \text{dacă } x \text{ e întreg} \\ 0, & \text{altfel} \end{cases}$

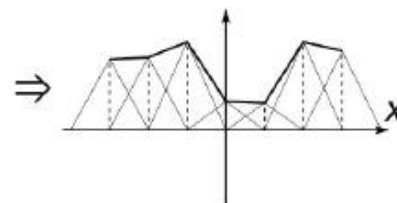
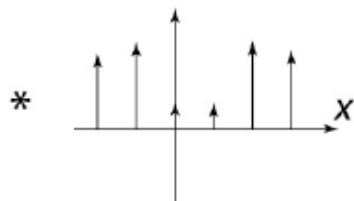
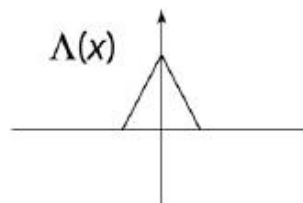
- reconstruim f prin convoluție cu un filtru de reconstrucție h

$$\tilde{f} = h * f_F$$

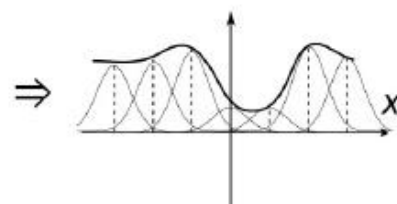
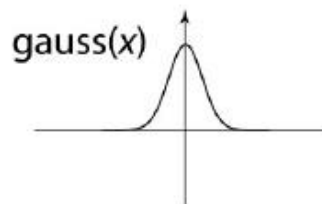
Tipuri de interpolare



Cel mai apropiat vecin



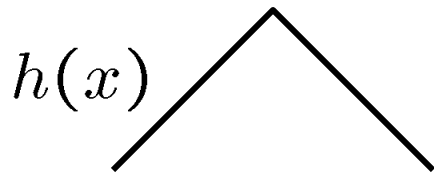
Liniară



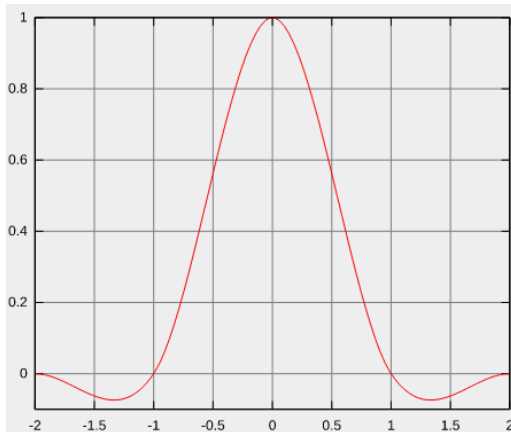
Gaussiană

Filtre de reconstrucție

- 1D 



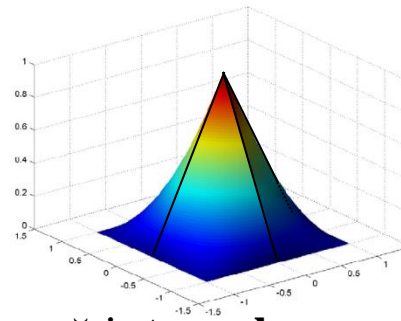
realizează **interpolare liniară**
pe baza celor mai apropiați 2 pixeli



realizează **interpolare cubică**
pe baza celor mai apropiați 4 pixeli

$h(x, y)$

2D



realizează **interpolarea biliniară**
pe baza celor
mai apropiați 4 pixeli

$$\begin{aligned} x_1 &= \lfloor x \rfloor & f_{11} &\equiv f(x_1, y_1) \\ x_2 &= \lfloor x \rfloor + 1 & f_{12} &\equiv f(x_1, y_2) \\ y_1 &= \lfloor y \rfloor & f_{21} &\equiv f(x_2, y_1) \\ y_2 &= \lfloor y \rfloor + 1 & f_{22} &\equiv f(x_2, y_2) \end{aligned}$$

$$f_{y1} = f_{11} + \frac{f_{21} - f_{11}}{x_2 - x_1}(x - x_1)$$

$$f_{y2} = f_{12} + \frac{f_{22} - f_{12}}{x_2 - x_1}(x - x_1)$$

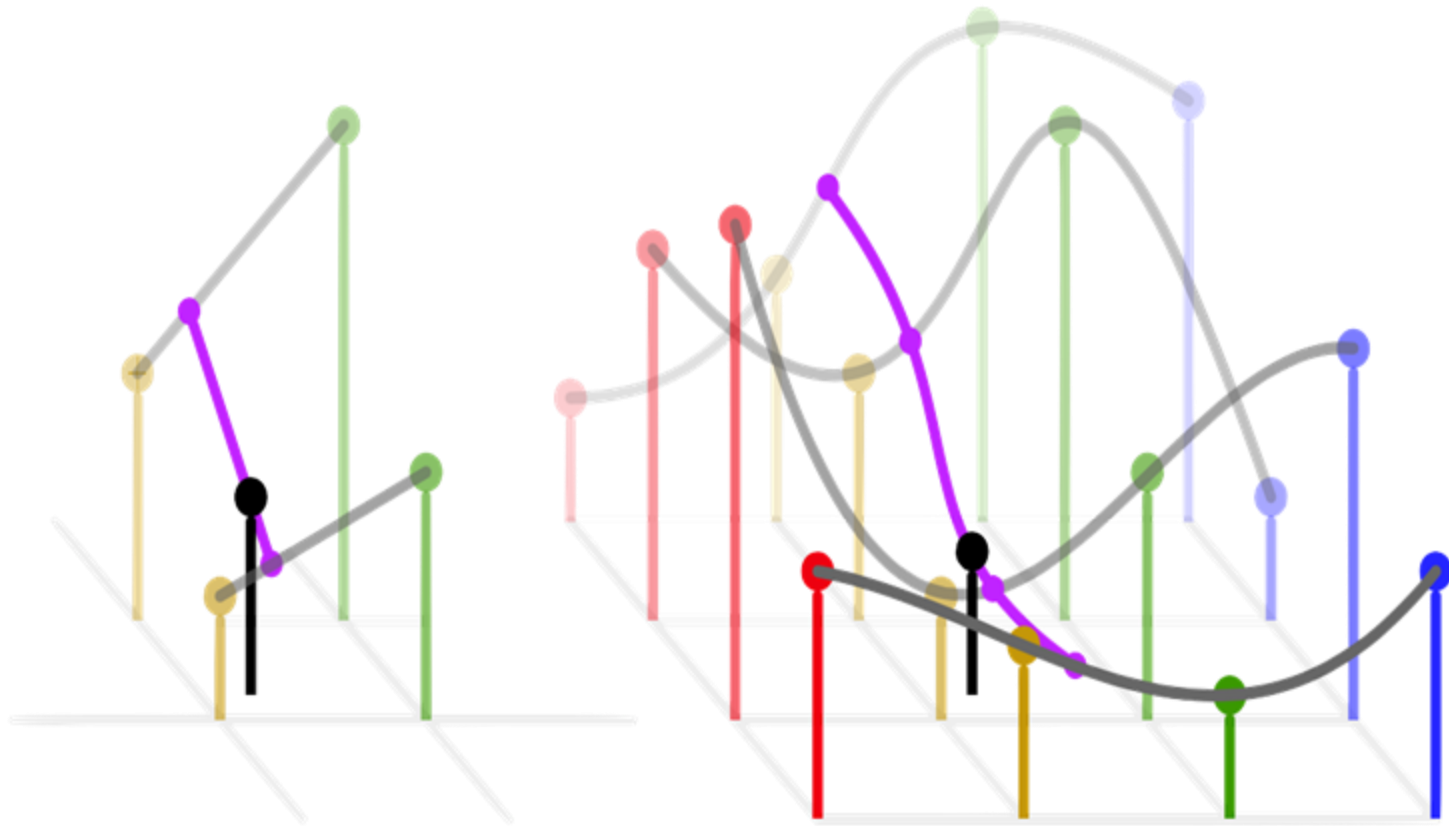
$$f(x, y) = f_{y1} + \frac{f_{y2} - f_{y1}}{y_2 - y_1}(y - y_1)$$

$$f(x; -1 \leq a < 0) = \begin{cases} (a+2)|x|^3 - (a+3)x^2 + 1 & \text{for } 0 \leq |x| \leq 1 \\ a|x|^3 - 5ax^2 + 8a|x| - 4a & \text{for } 1 < |x| \leq 2 \\ 0 & \text{otherwise} \end{cases}$$

2D 

realizează **interpolare bicubică**
pe baza celor mai apropiați 16 pixeli

Filtre de reconstrucție



Bilinear

Bicubic

Interpolarea imaginilor

Imagine inițială:  x 10



Interpolare “cel mai apropiat vecin”



Interpolare biliniară

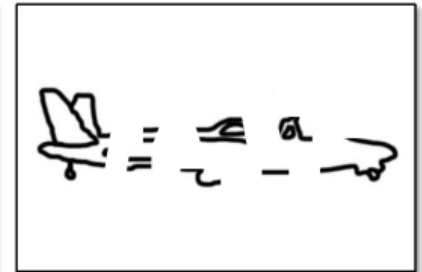
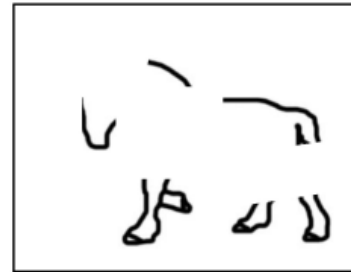


Interpolare bicubică

Gradienți & muchii

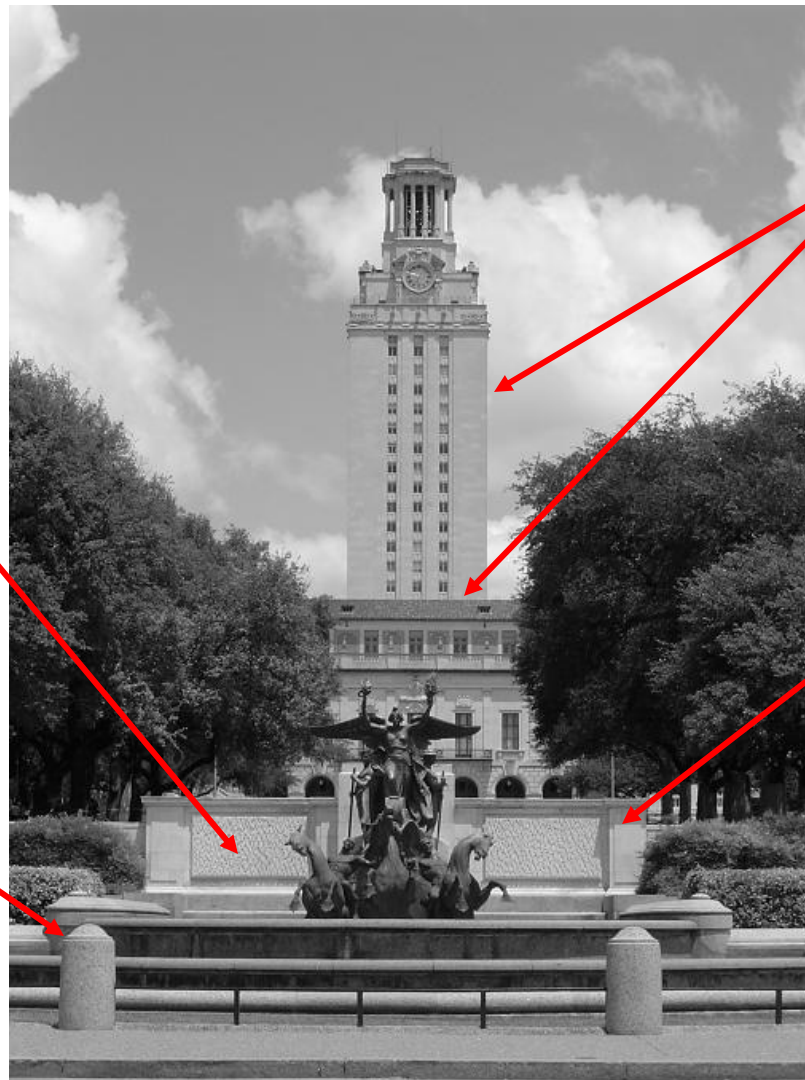
Detectarea muchiilor

- **Scop:** transformăm imaginea dintr-o matrice de pixeli într-o mulțime de curbe/segmente/contururi
- **De ce?**



- **Idee principală:** localizare gradienti, post-procesare

Cum se formează muchiile?



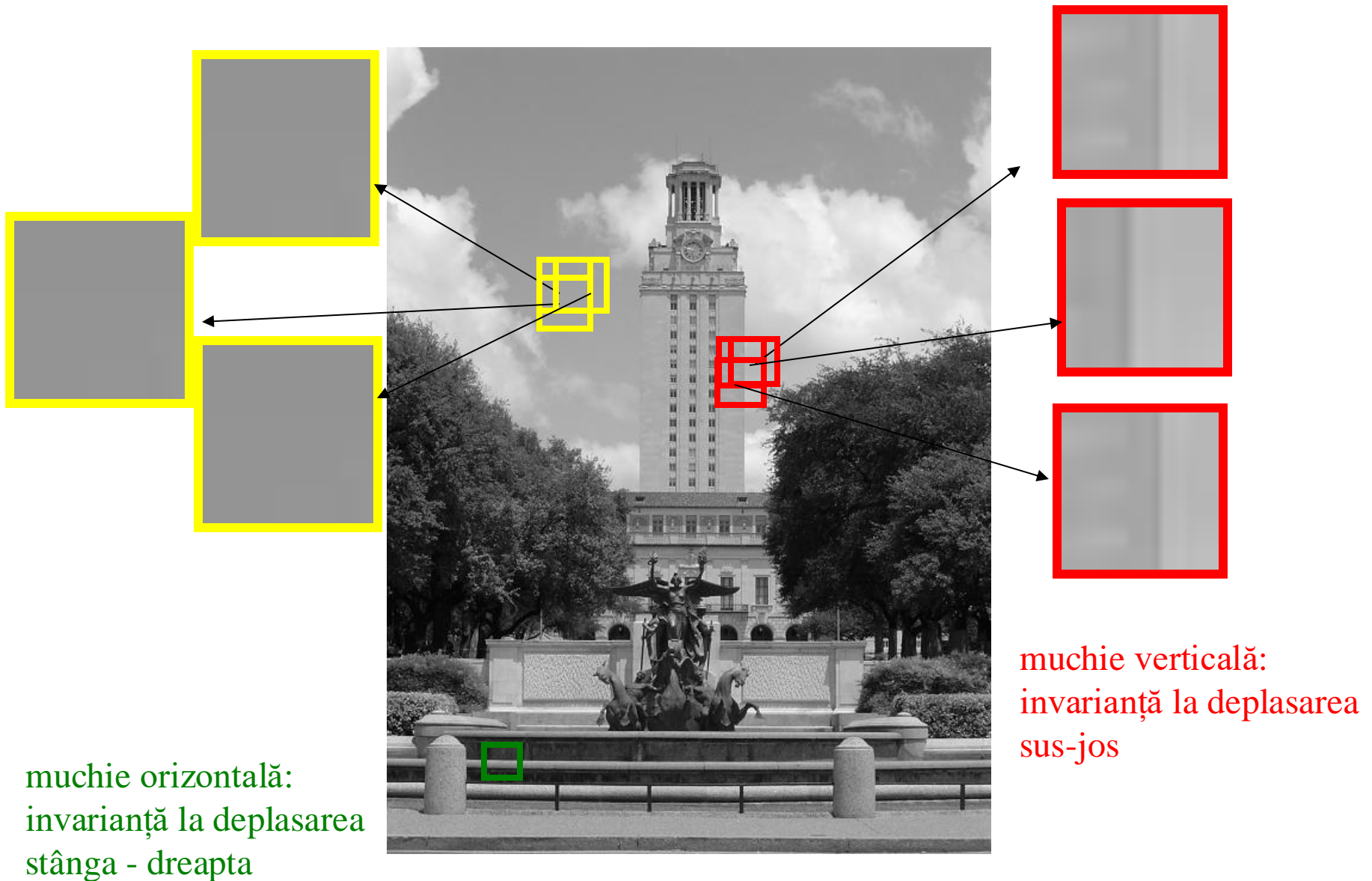
Reflexia luminii în
funcție de textură

Discontinuități în 3D:
conturul obiectelor

Formarea umbrelor

Discontinuitatea în
orientarea suprafeței

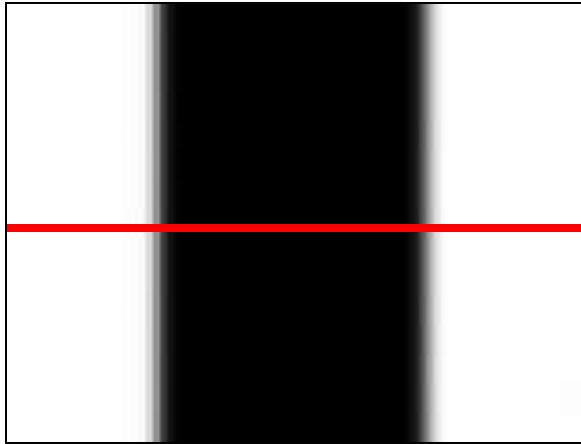
Muchii/gradienti și invarianță



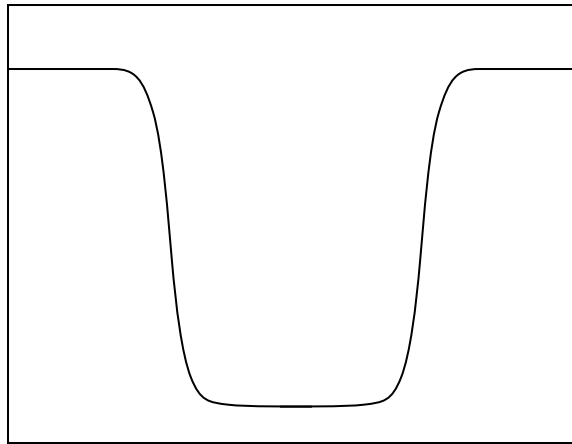
Derivate și muchii

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

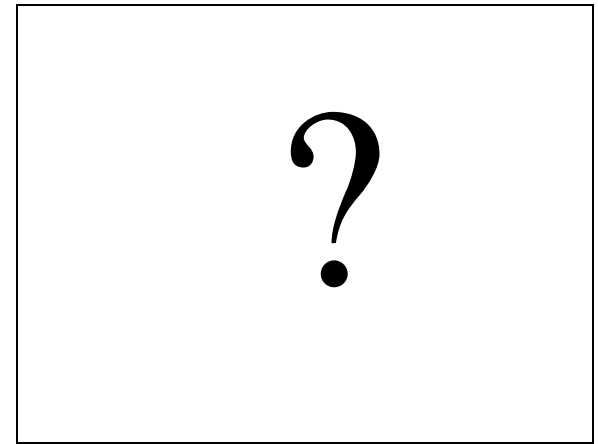
image



funcția de intensitate
(de-a lungul liniei roșii)

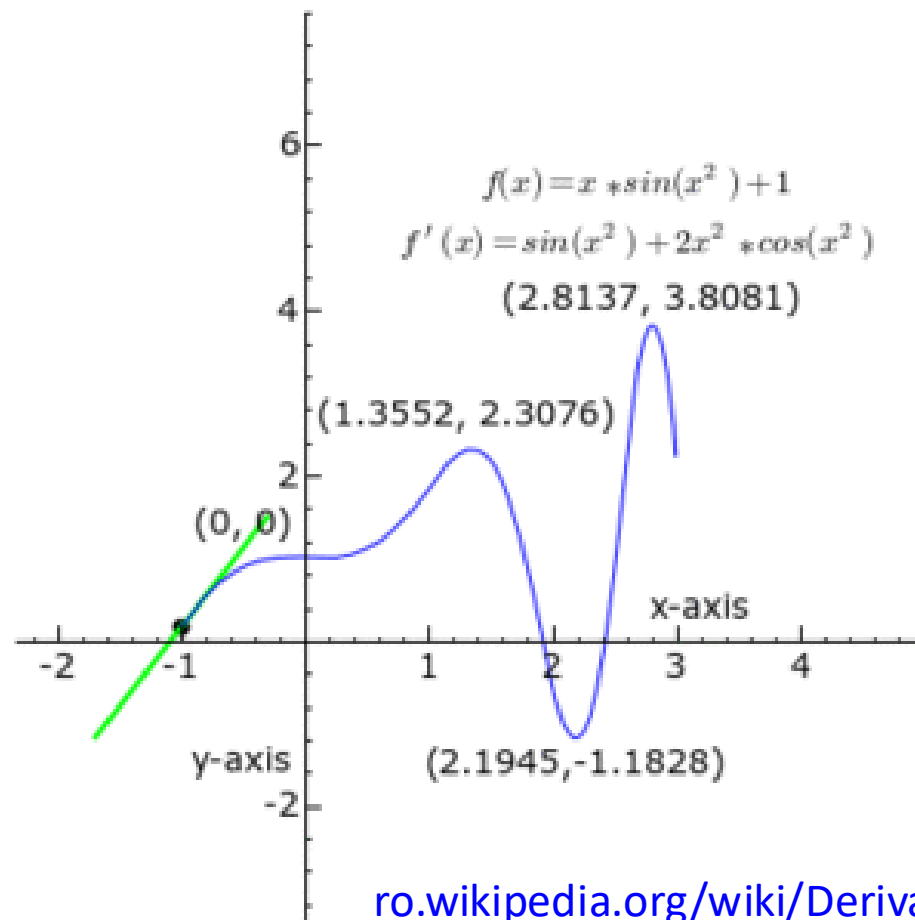


derivata funcției
de intensitate



Derivata unei funcții

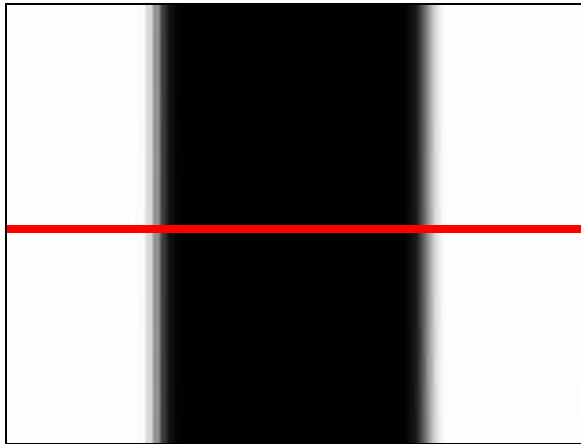
- panta (înclinarea) dreptei tangente la grafic
- **tangenta verde > 0**
- **tangenta neagră $= 0$**
- **tangenta roșie < 0**



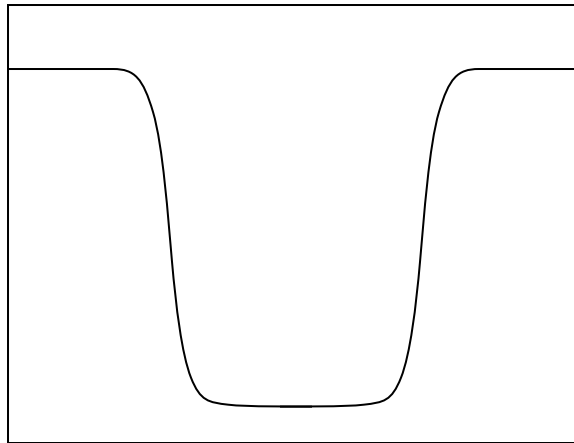
Derivate și muchii

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

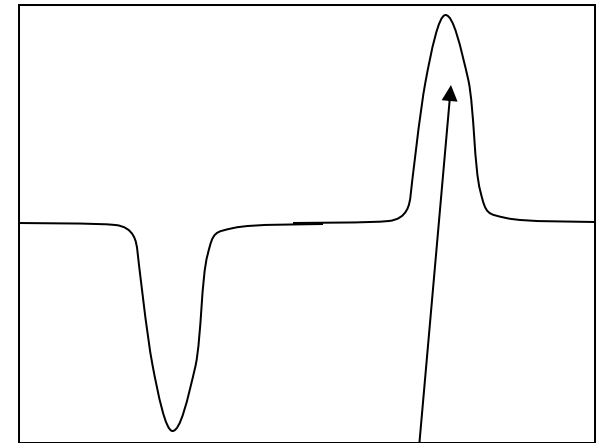
image



funcția de intensitate
(de-a lungul liniei roșii)



derivata funcției
de intensitate



muchiile corespund punctelor
de extrem ale derivatei

Calculul derivatelor prin corelație/convoluție

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

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

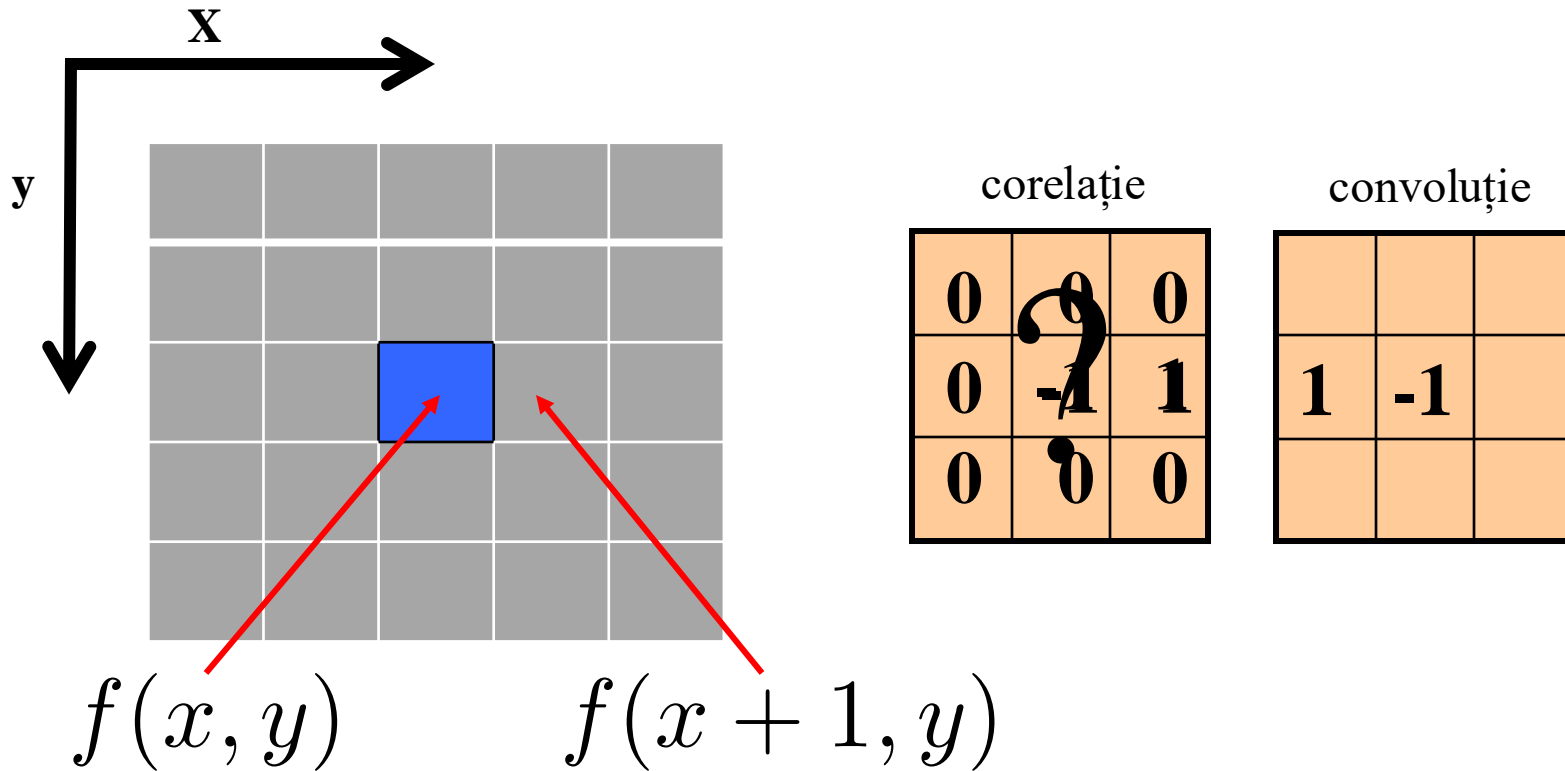
Pentru cazul discret (**cazul imaginilor**), putem aproxima derivata folosind diferențe finite:

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

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

Calculul derivatelor prin corelație/convoluție

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

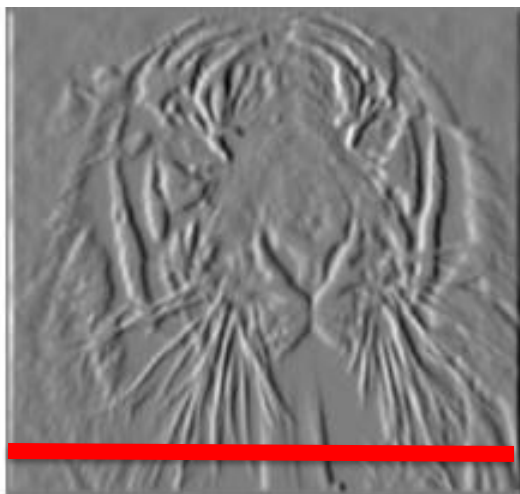


Derivatele parțiale ale unei imagini



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

-1	1
----	---



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

?

-1
1

sau

1
-1

Care imagine arată schimbările în raport cu x/y ?

(arătăm filtrele pentru corelație)

Slide adaptat după S. Lazebnik

Filtre Sobel

Există alte aproximări (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: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

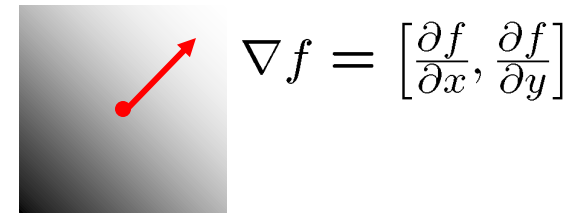
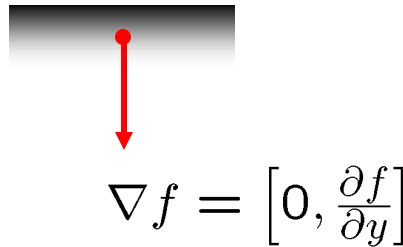
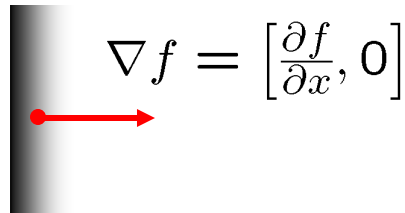
Roberts: $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

```
img = cv2.imread('simona.jpg',0)
sobelx = cv2.Sobel(img,cv2.CV_64F,1,0,ksize=5)
plt.imshow(sobelx,cmap = 'gray')
plt.show()
```



Gradientul unei imagini

Gradientul unei imagini în fiecare punct: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$



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 mărimea 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:

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

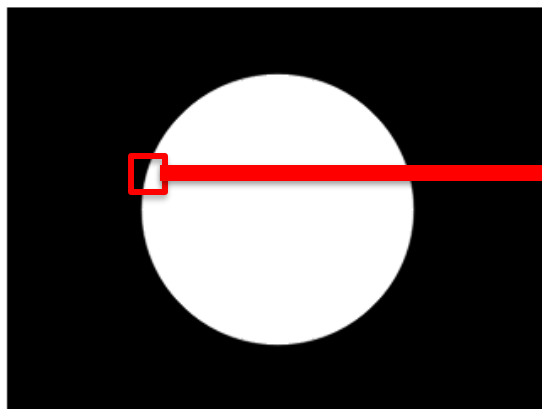
Calculul gradientului unei imagini

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

folosesc filtrul
Sobel vertical

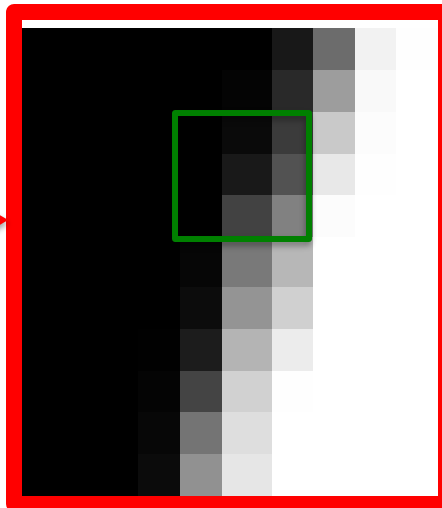
$M_x =$

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



imaginea f

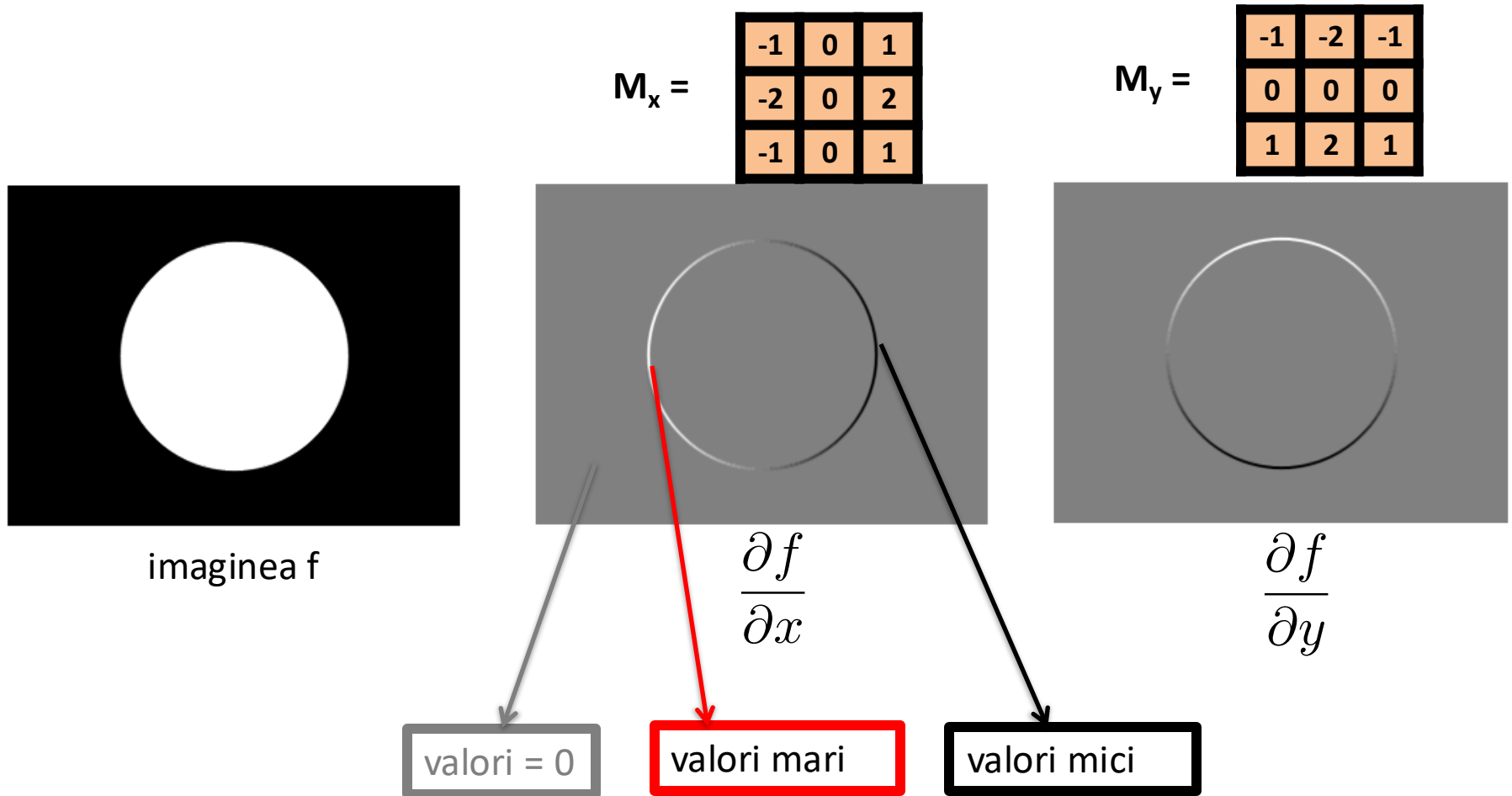
zoom



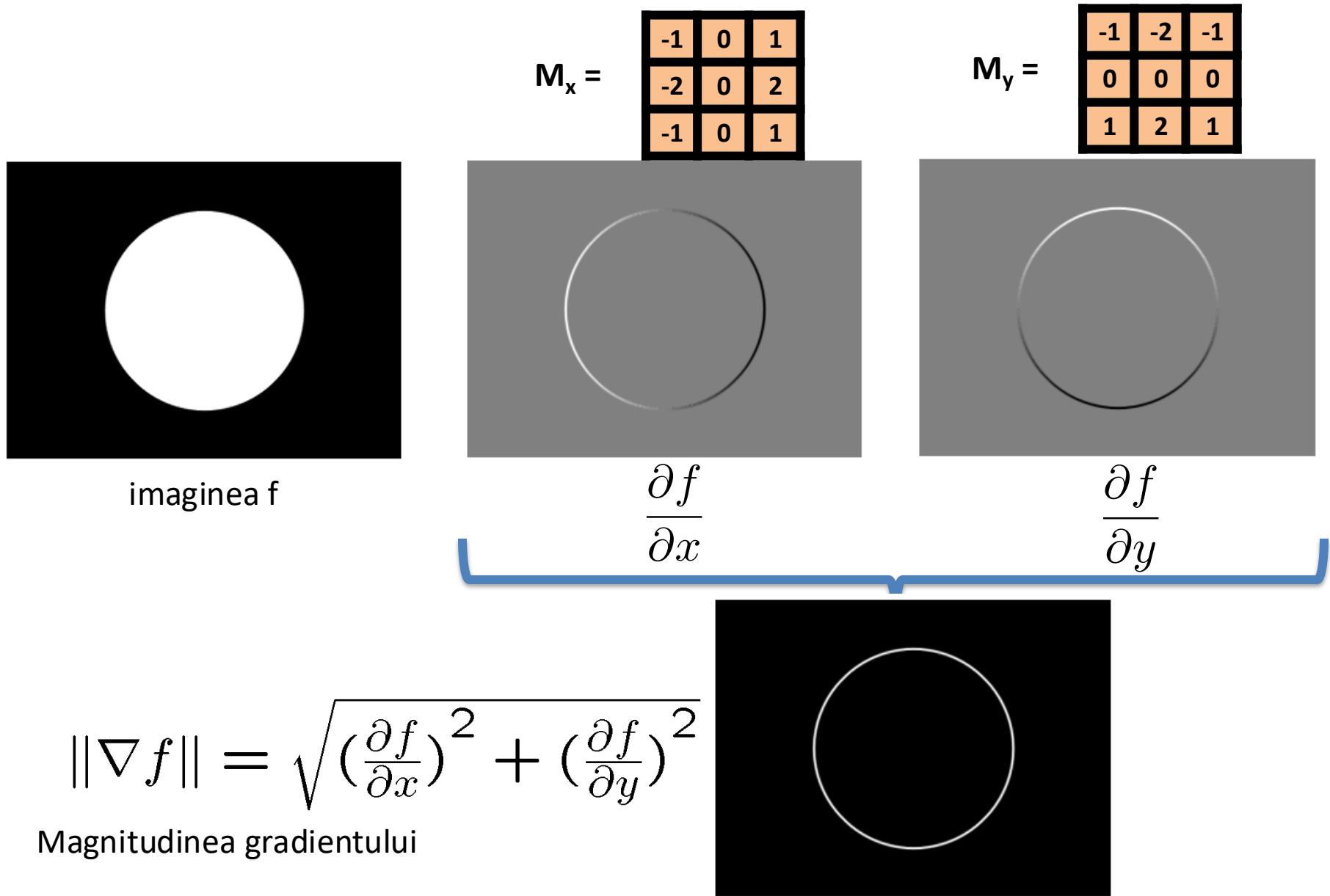
0	0	0	0	4	41	157	249	2
0	0	0	0	10	59	201	252	2
0	0	0	0	25	82	232	254	2
0	0	0	0	66	129	252	255	2
0	0	0	6	121	183	255	255	2
0	0	0	1	148	208	255	255	2
0	0	1	2	180	236	255	255	2
0	0	4	6	209	254	255	255	2
0	0	7	11	222	255	255	255	2

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

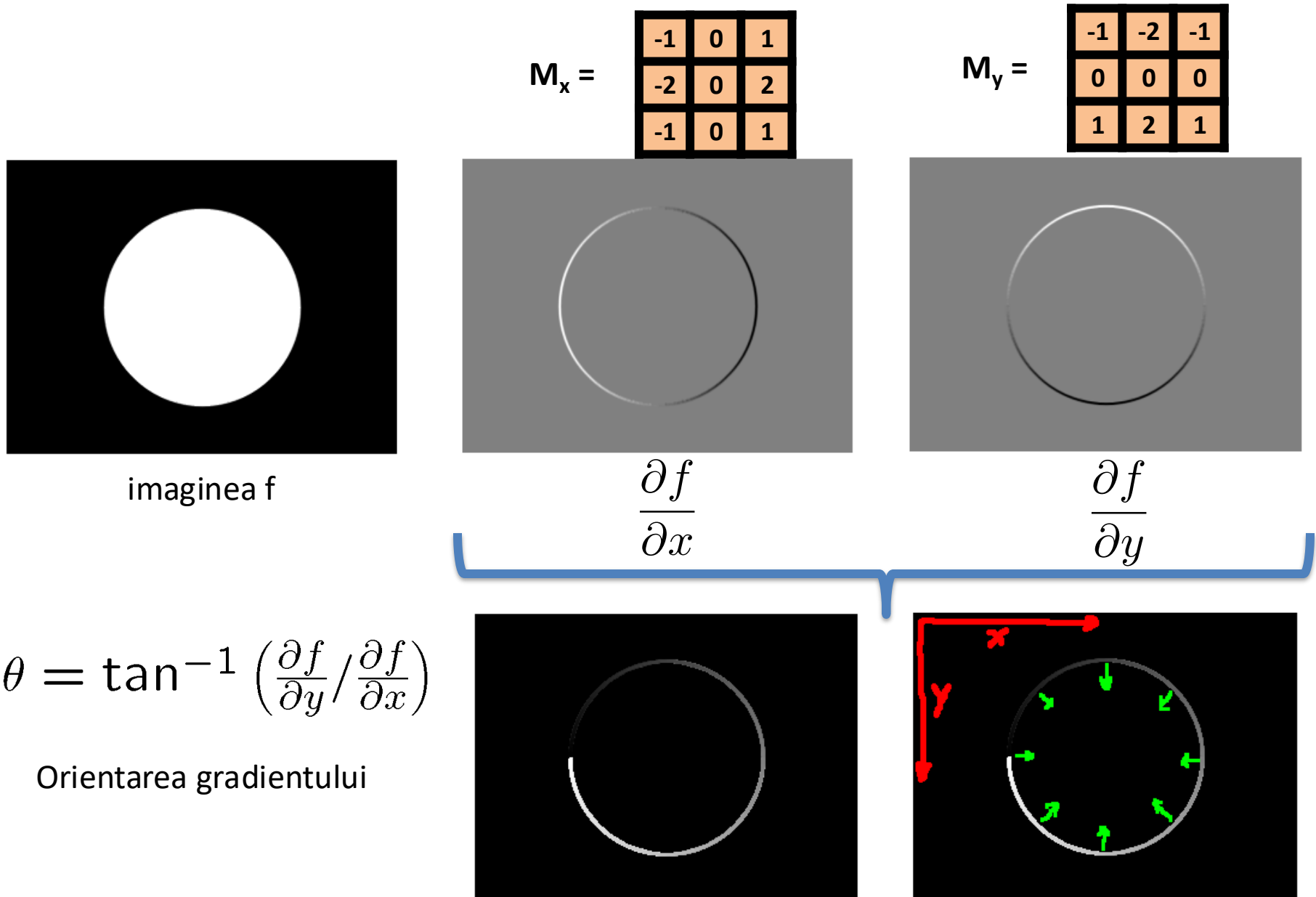
Calculul gradientului unei imagini



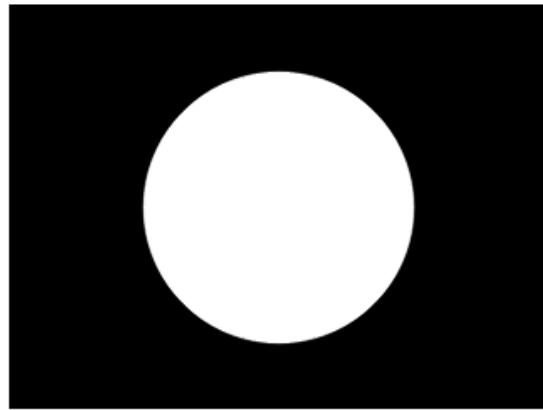
Calculul gradientului unei imagini



Calculul gradientului unei imagini



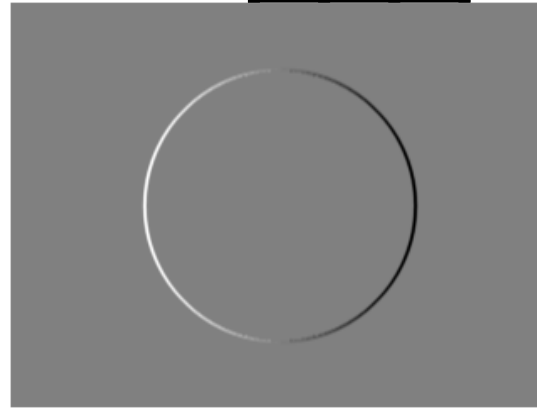
Calculul gradientului unei imagini



imaginea f

$M_x =$

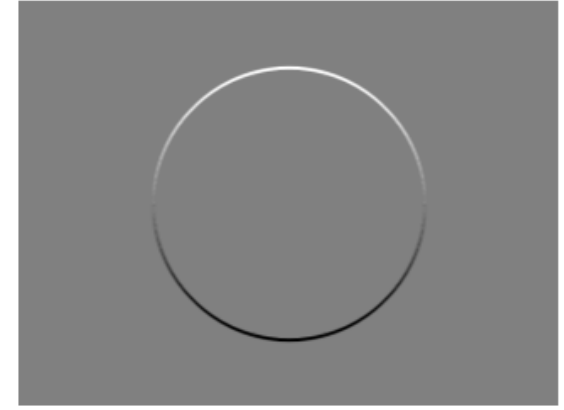
-1	0	1
-2	0	2
-1	0	1



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

$M_y =$

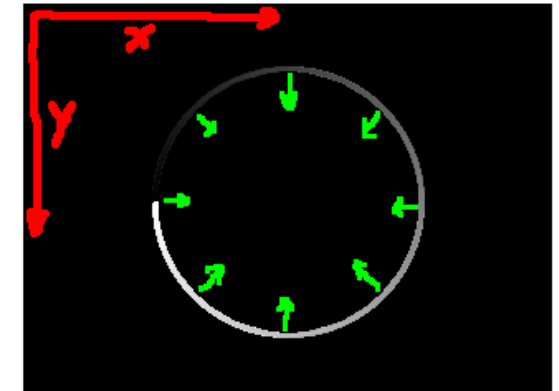
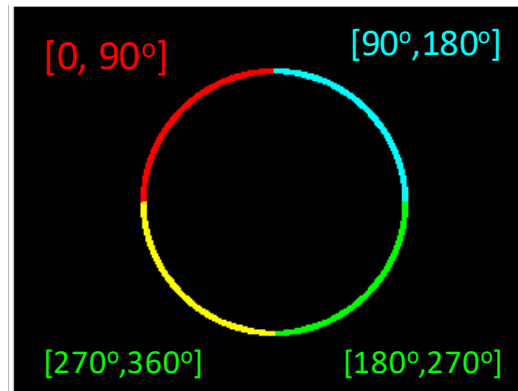
-1	-2	-1
0	0	0
1	2	1



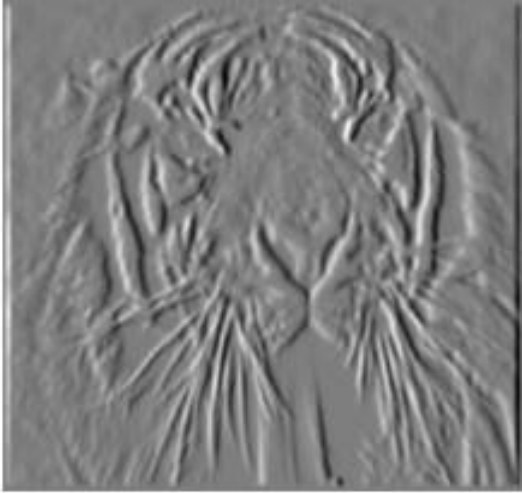
$\frac{\partial f}{\partial y}$

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

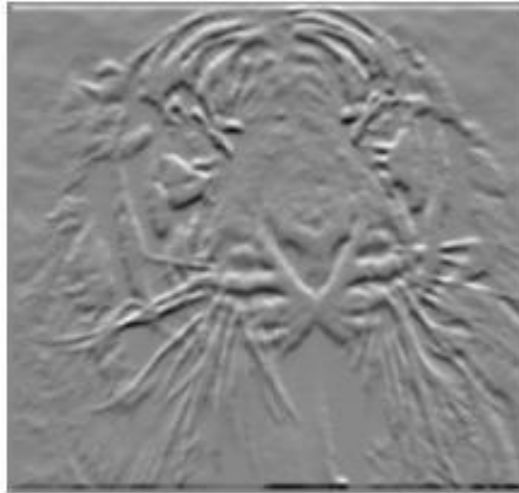
$[0, 90^\circ]$
 $[90^\circ, 180^\circ]$
 $[180^\circ, 270^\circ]$
 $[270^\circ, 360^\circ]$



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



image



Magnitudinea gradientului

Aplicație laborator:
Redimensionarea imaginilor cu
păstrarea conținutului