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

Detalii organizatorice

1. Recuperare laborator săptămâna 1: vinerea aceasta 10-12 + 12-14 e ok pentru voi? Alternativ luni 8-10 + 10-12

2. De adăugat o grupă de laborator? (sunteți 100 de studenți)

Cuprinsul trecut

1. Aspecte organizatorice legate de cursul de VA
2. Ce este VA?
3. Aplicații de succes în VA
4. Formarea imaginilor
5. Structura cursului de VA
6. Bibiliografie
7. Kahoot!

Examen – evaluare în iarnă

- Nota finală = min(10, max(T1 + T2 + min(1, BC + BL), 2*L1 + min(1, BC), L1 + max(T1,T2) + min(1, BC)))

- nu există praguri minimale de îndeplinit

Restanță + reexaminare + mărire - evaluare

- puteți obține nota numai din lucrarea finală de laborator (test pe calculator în sesiune) + bonusul de la curs.
  - nota N4 = 2*L2 + min(1, BC)

SAU

- puteți obține nota din lucrarea finală de laborator (test pe calculator în sesiune) și o notă din temă (trebuie să aveți cel puțin o temă rezolvată) + bonusul de la curs.
  - nota N5 = L2 + max(T1,T2) + min(1, BC)

- Nota finală = min(10, max(N4, N5))

Bonusul de la curs + laborator

- puteți acumula maxim 1 punct bonus de la curs + laborator;
- bonus de la laborator = 0,05p pentru fiecare prezență la laborator (maxim o prezență pe săptămână);
- bonusul de la curs:
  - la sfârșitul fiecărui curs veți primi întrebări din materia predată la acel curs;
  - vom folosi platforma Kahoot!
  - număr de întrebări flexibil în funcție de fiecare curs;
  - primiți puncte dacă răspundeți correct și rapid;
  - premiem la fiecare curs primii 12 studenți din clasamentul asociat : primii 3 studenți iau 0.2p, următorii 3 studenți iau 0.15p, următorii 3 studenți iau 0.1p, următorii 3 studenți iau 0.05p ;

Temele de anul trecut

https://tinyurl.com/CAVA-2023-TEMA1

https://tinyurl.com/CAVA-2023-TEMA2

https://tinyurl.com/CAVA-2022-TEMA1

https://tinyurl.com/CAVA-2022-TEMA2

https://tinyurl.com/CAVA-2021-TEMA1

https://tinyurl.com/CAVA-2021-TEMA2

Tipuri de imagini digitale

Binare

Grayscale
(tonuri de gri)

Color

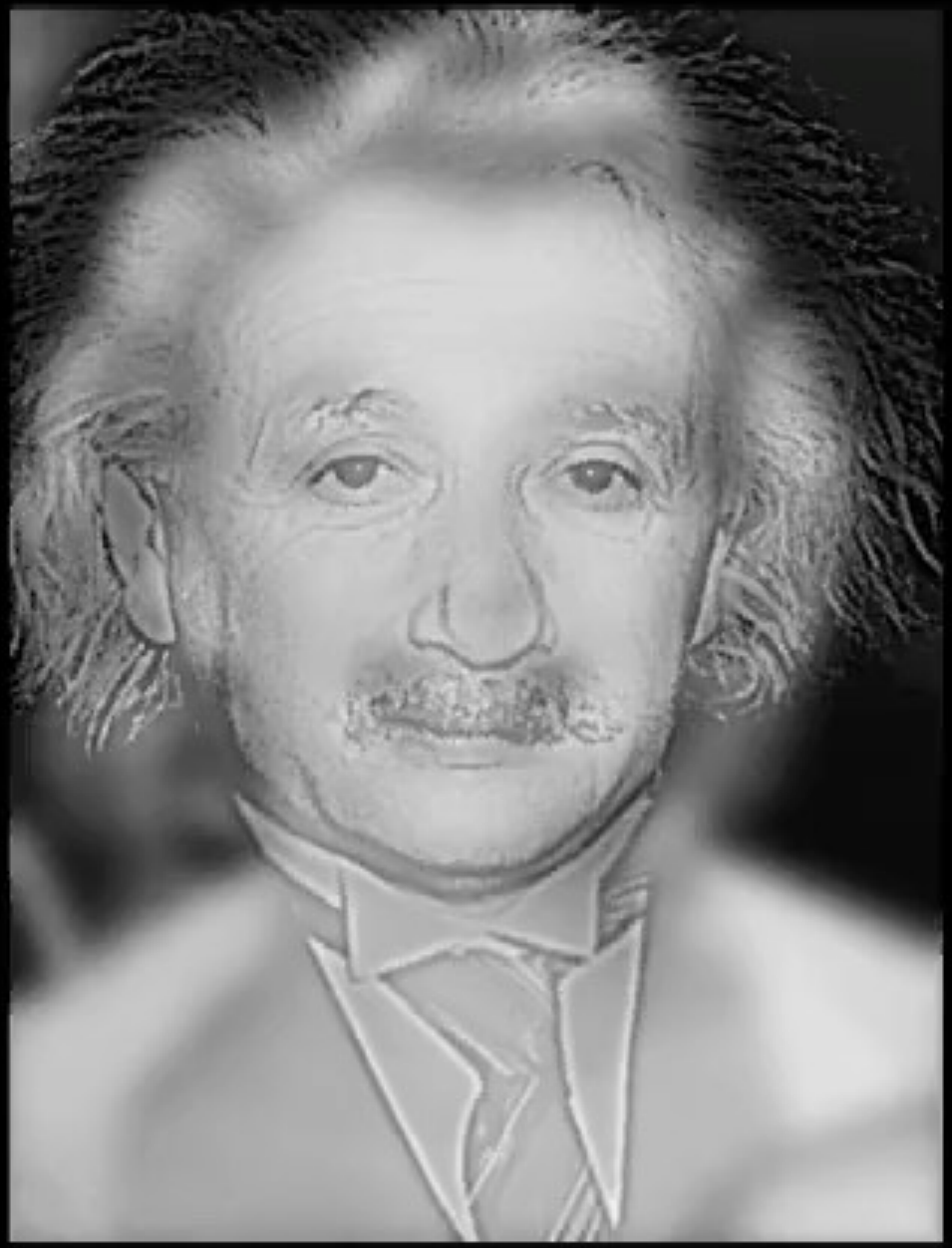
Luminozitate	**negru, alb**	**tonuri de gri**	**R G B**
Valori	{**0**,**1**}	{**0**, … , **255**}	{0, …,255}^{3}
Culori	**negru - 0**, **alb - 1**	**negru - 0**, **gri - 128**, **alb - 255**	**(255,0,0)**, **(0,255,0)**, **(0,0,255)**, **(0,0,0)**,**(255,255,255)**, **(255,255,0)**, **(255,125,0)**, **(0,255,255)**, **(255,0,255)**
Memorie/pixel	1 bit/pixel	8 biți/pixel	24 biți/pixel

Cursul de azi

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

normal

- Filtrarea neliniară
  - filtrul median



Laborator 2+3: Imagini mozaic

imagine de referință

+

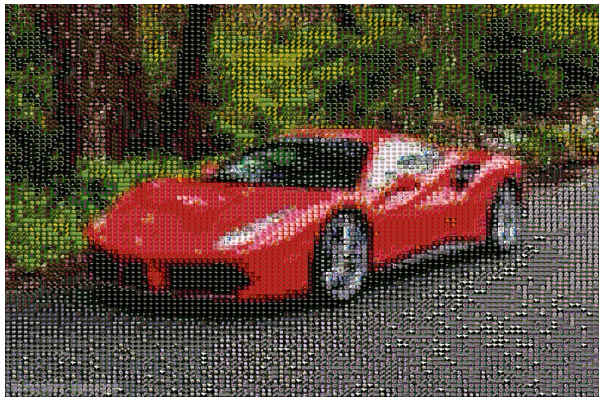
colectie de imagini (piese)
de dimensiuni reduse
28×40 pixeli

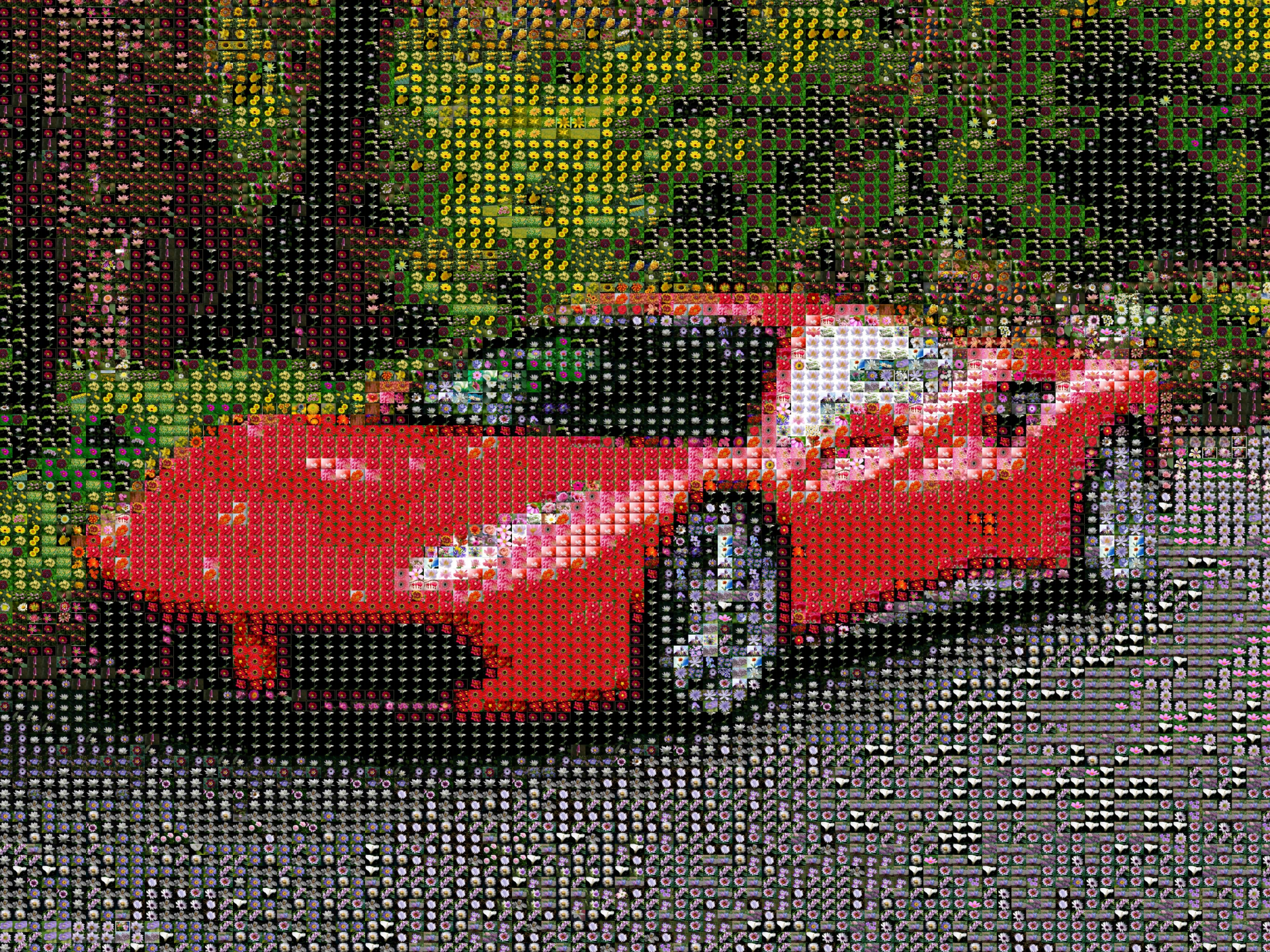
+

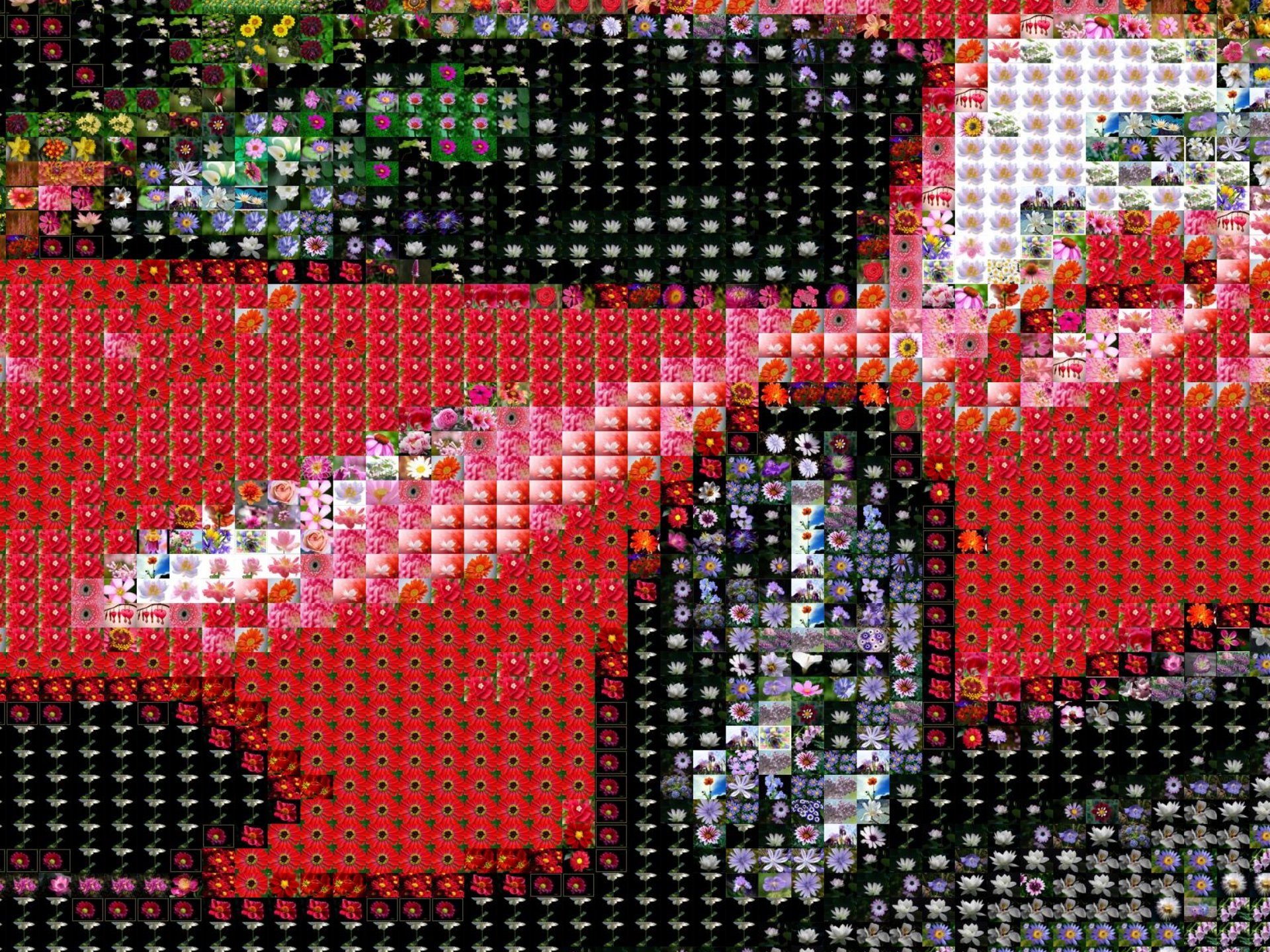
algoritm codat de voi

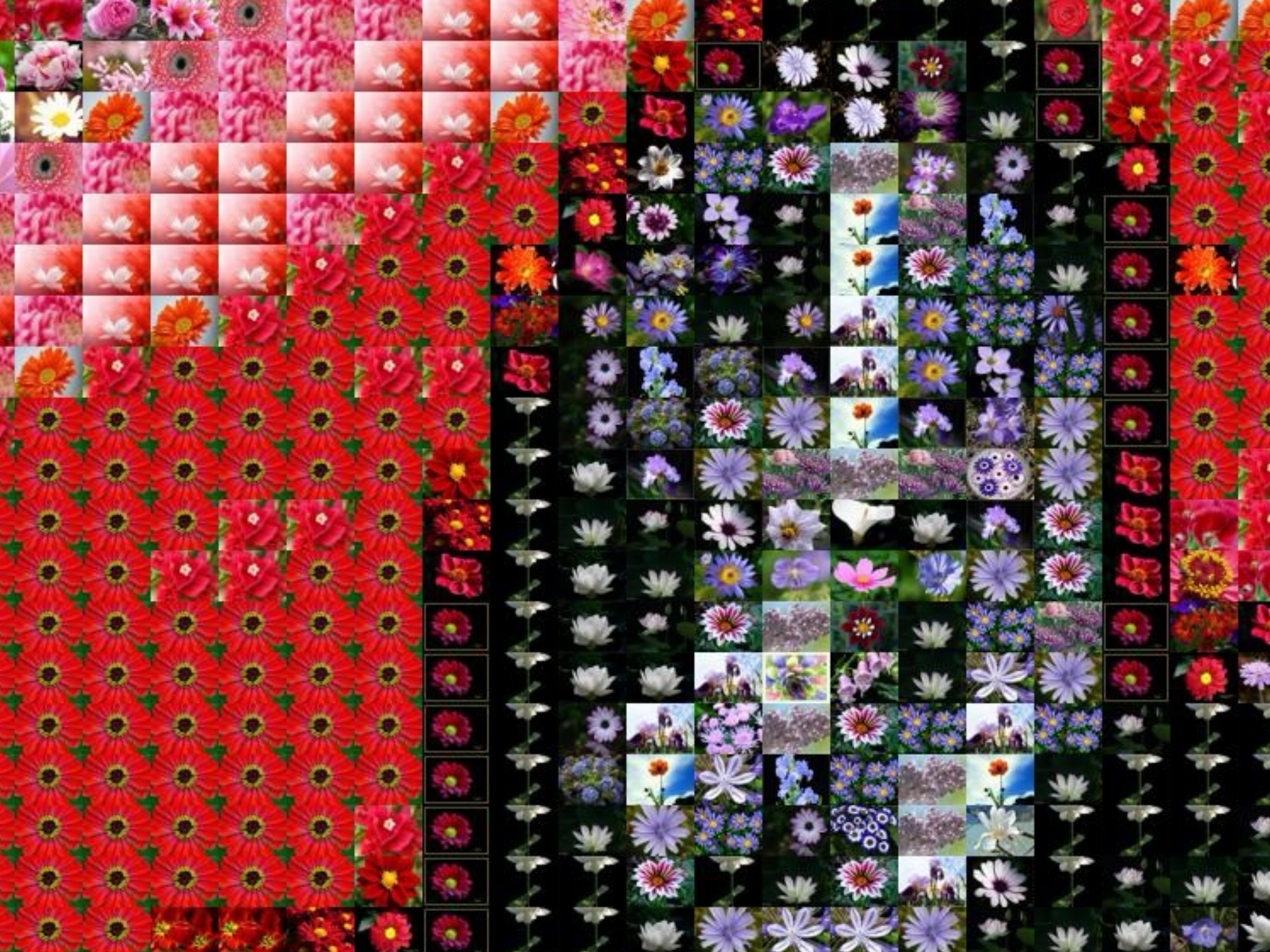
=

imagine mozaic









Algoritm

Input:

1. imagine de referință

2. colecție de imagini (piese)
   500 de piese a 28 x 40 pixeli

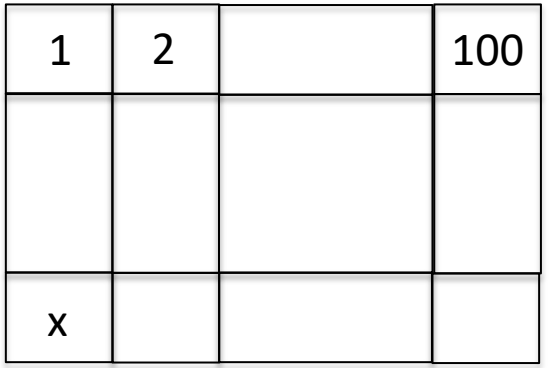
3. număr piese în lățime ale mozaicului (implicit = 100)

1	2		100

Algoritm – varianta 1

Pași:

1. determină înălțimea mozaicului:
   - păstrează proporția (aspect ratio) imaginii de referință inițiale
   - multiplu de înălțimea unei piese



2. adaugă piese în mozaic pe un grid (caroiaj): aplică un criteriu pentru a selecta piesa care se potrivește cel mai bine într-o poziție:
   - aleator (alegem piesele întâmplător)
   - culoarea medie cea mai apropiată (cea mai mica distanță euclidiană)

Criterii

- aleator: alegem piesele la întâmplare, considerând că orice piesă la fel de bună ca cealaltă în orice poziție

imagine de referință

imagine mozaic

Criterii

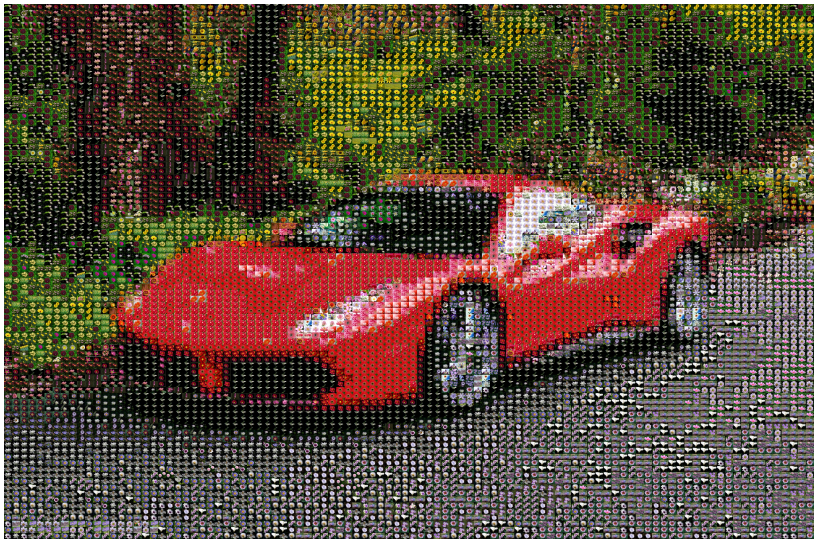
- culoarea medie cea mai apropiată:
  - calculăm culoarea medie a pixelilor din blocul de înlocuit din imaginea de referință redimensionată;
  - culoarea medie este un triplet (r_{medie}, g_{medie}, b_{medie}) pentru imagini color sau un scalar i_{medie} pentru imagini gri;
  - calculăm culoarea medie a pixelilor din fiecare piesă;
  - calculăm cea mai apropiată (distanța euclidiană cea mai mică) culoare medie a unui piese de culoarea medie a blocului;
  - înlocuim blocul cu piesa aleasă.

Criterii

- culoarea medie cea mai apropiată

imagine de referință

imagine mozaic

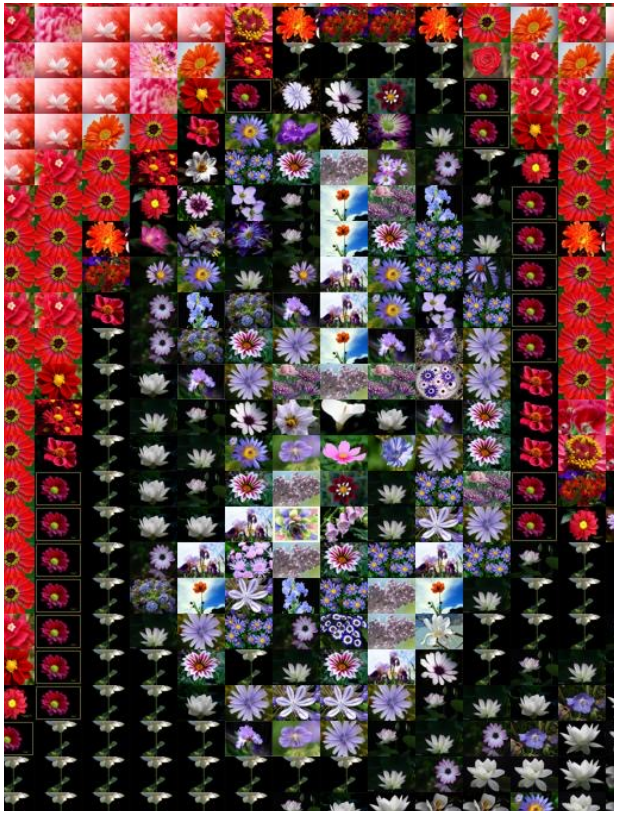


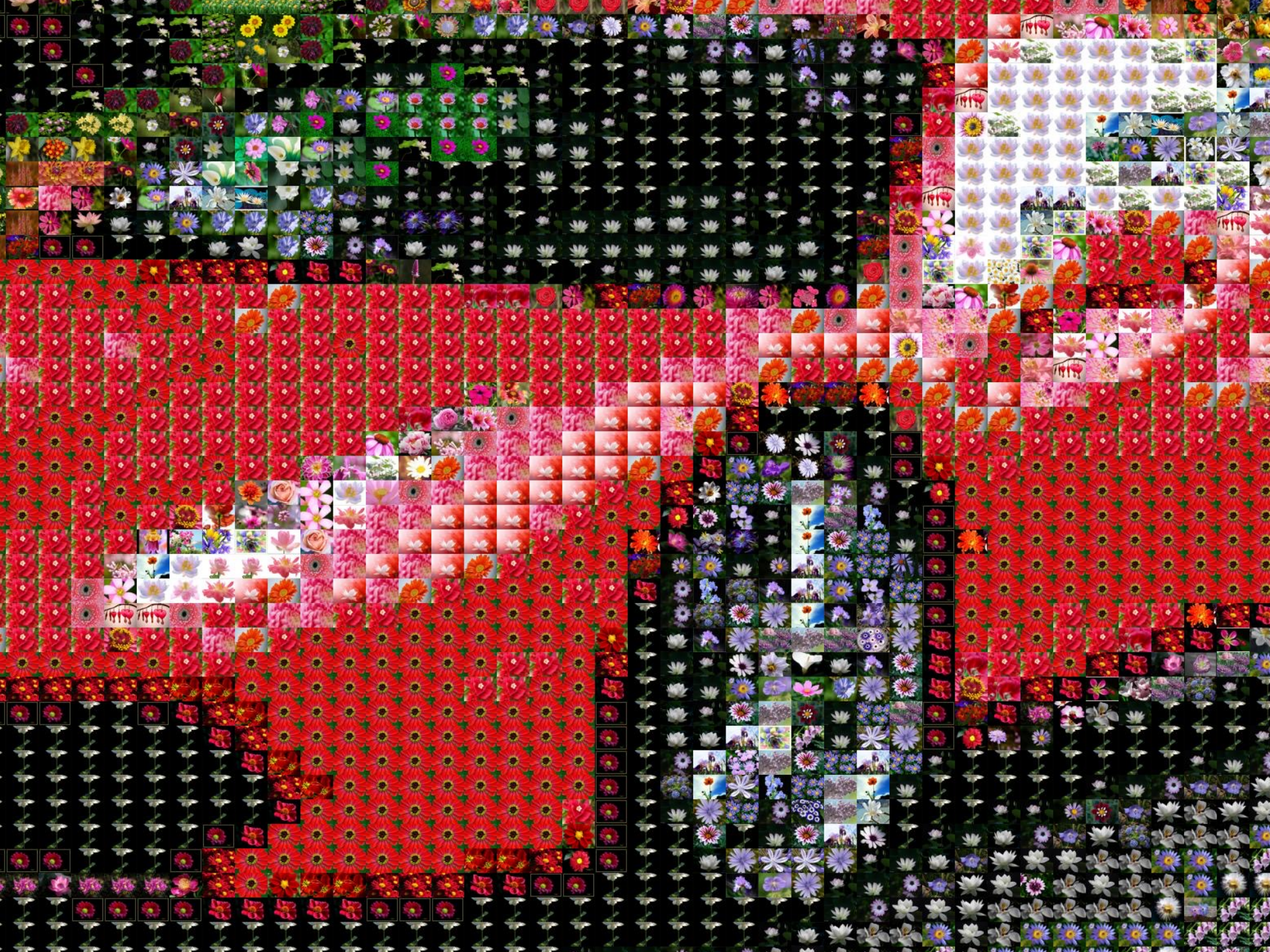
Criterii

- culoarea medie cea mai apropiată

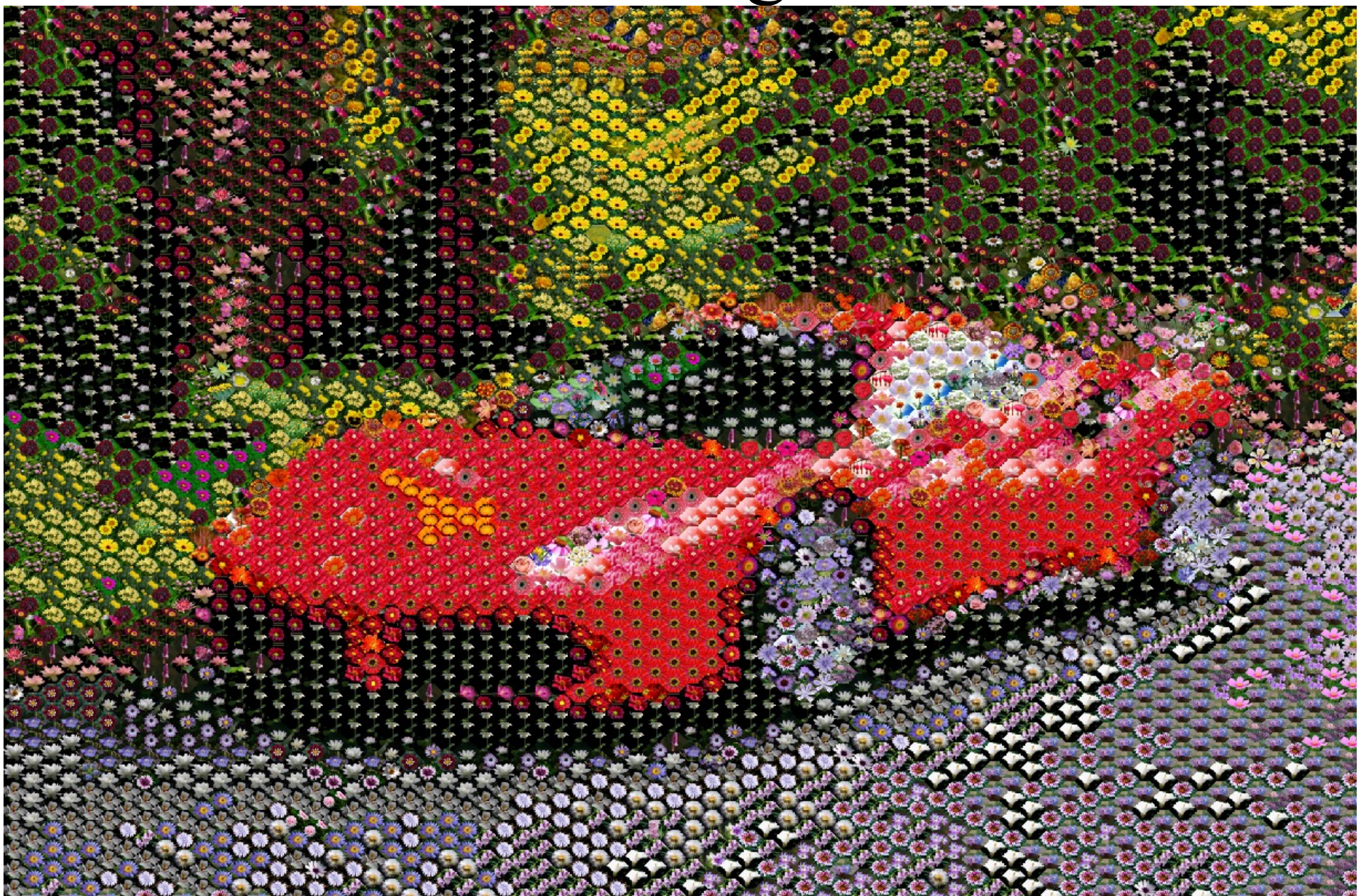
imagine de referință

imagine mozaic

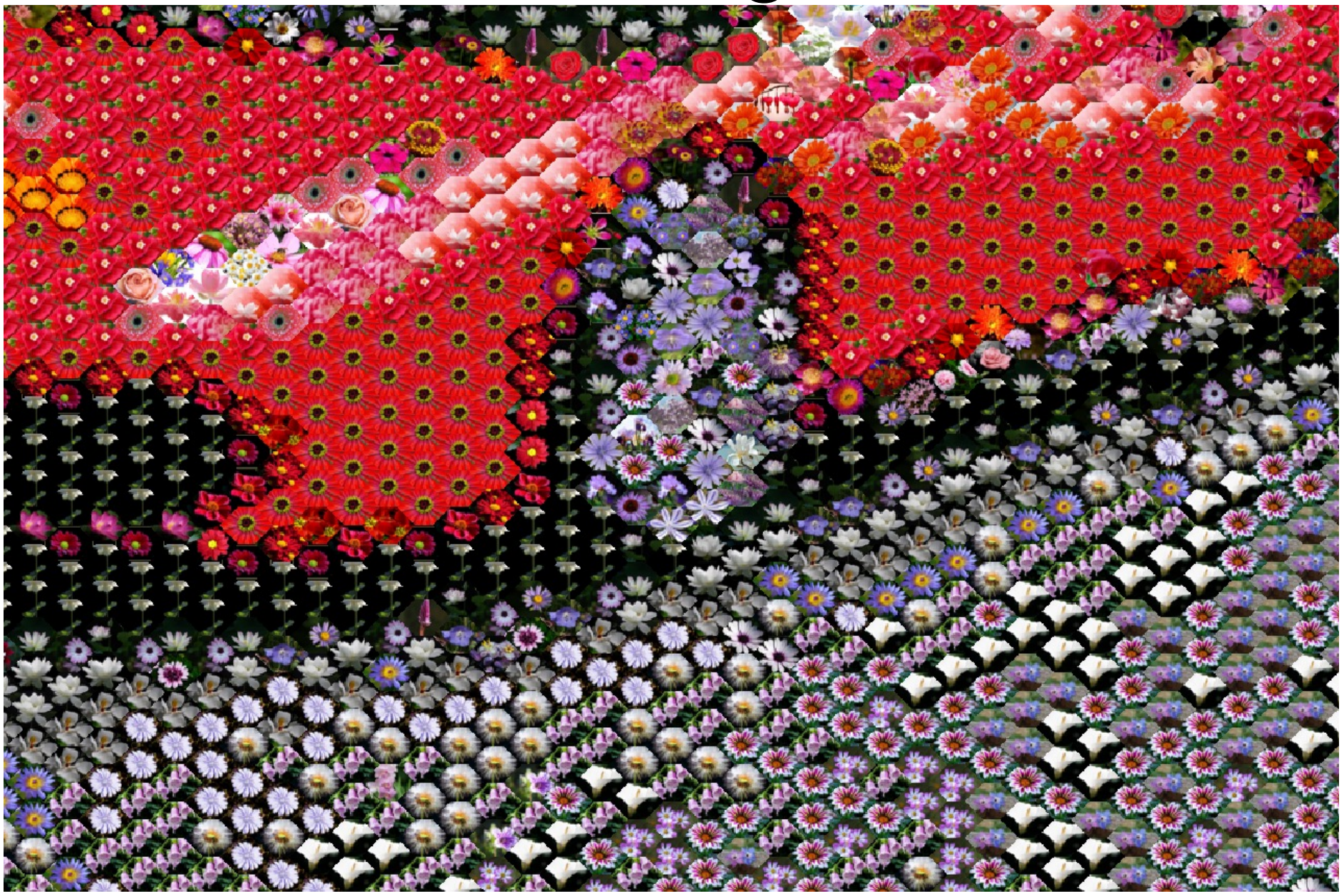




Piese hexagonale



Piese hexagonale



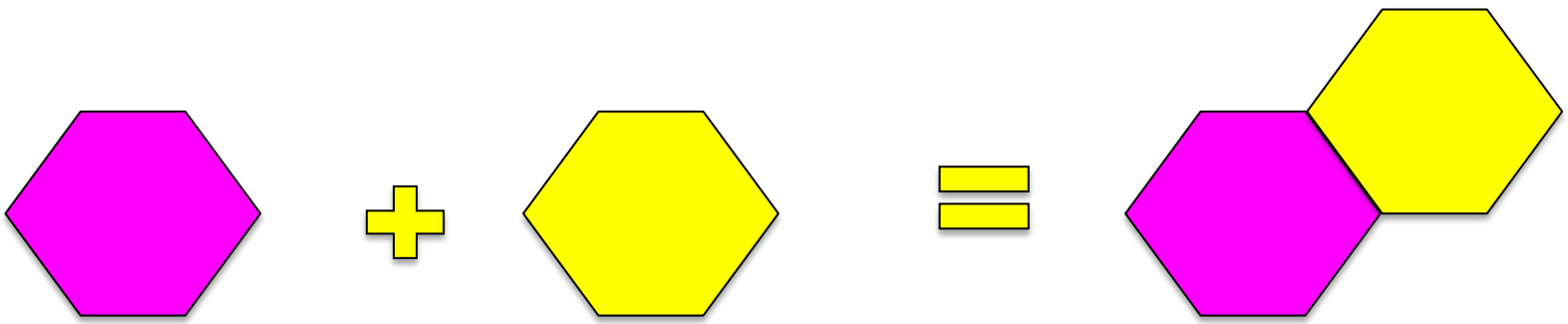
Piese hexagonale

Piese hexagonale

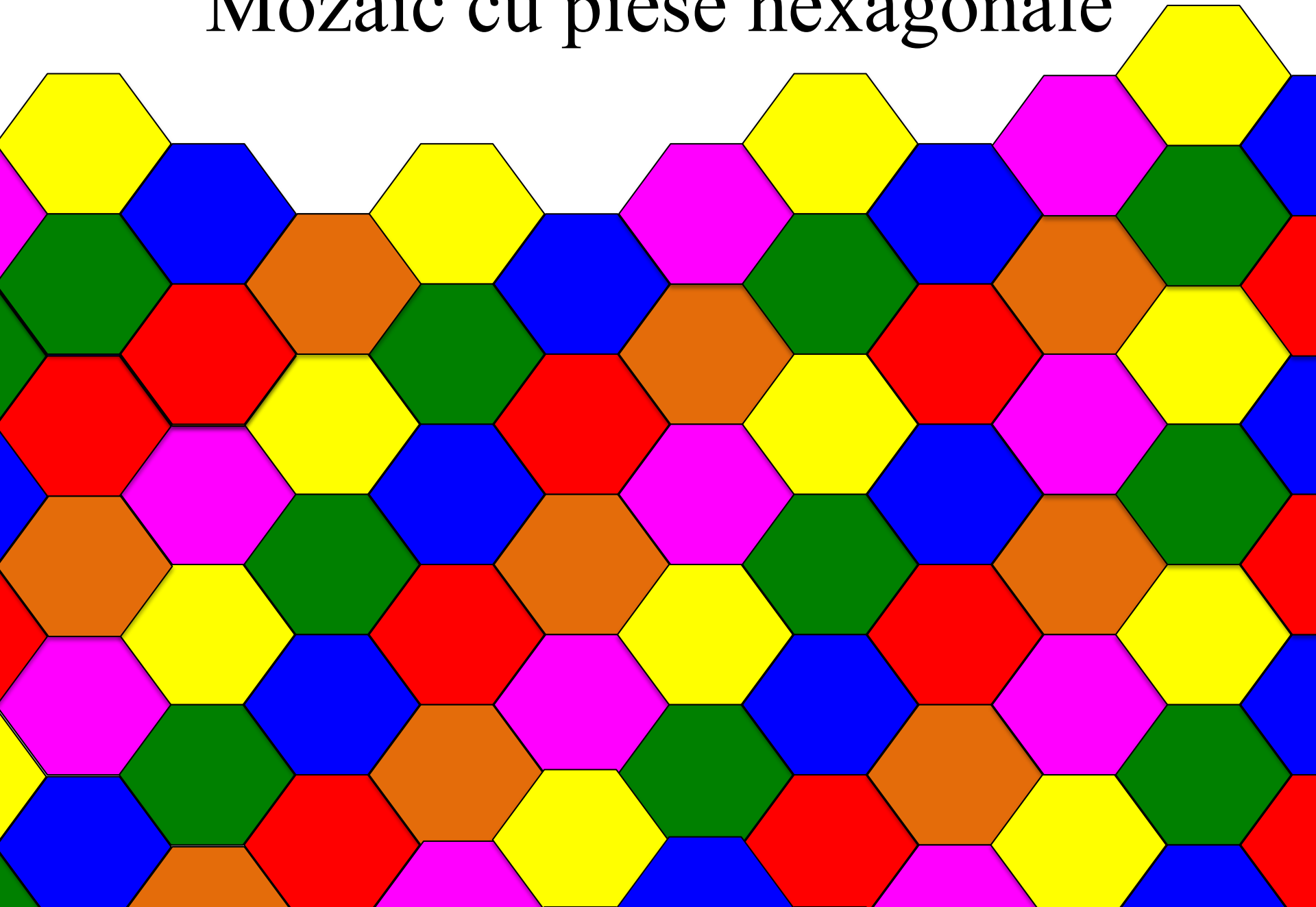
Piesă inițială

Mască (valori de 0 și 1)

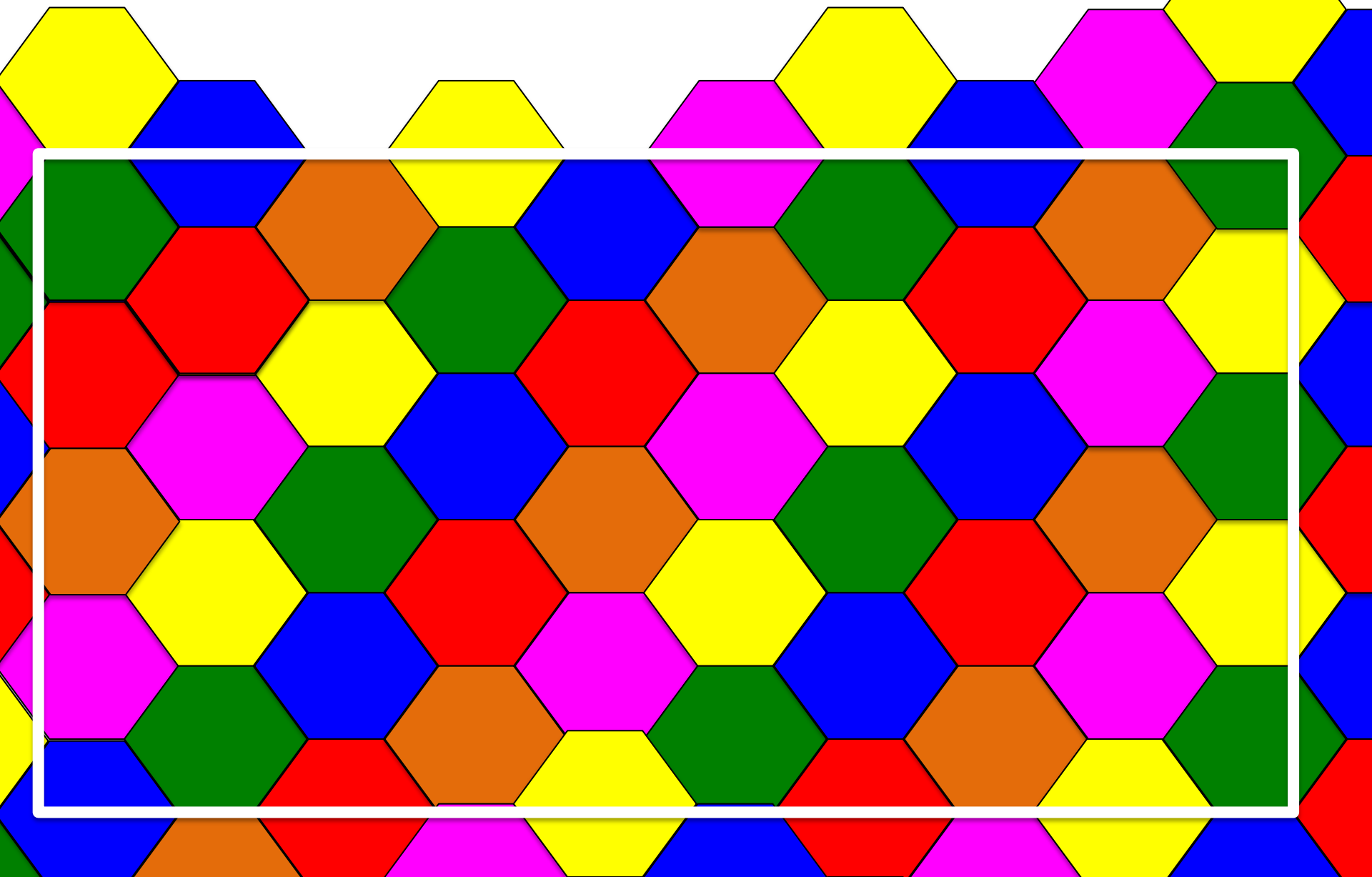
Piesă hexagonală



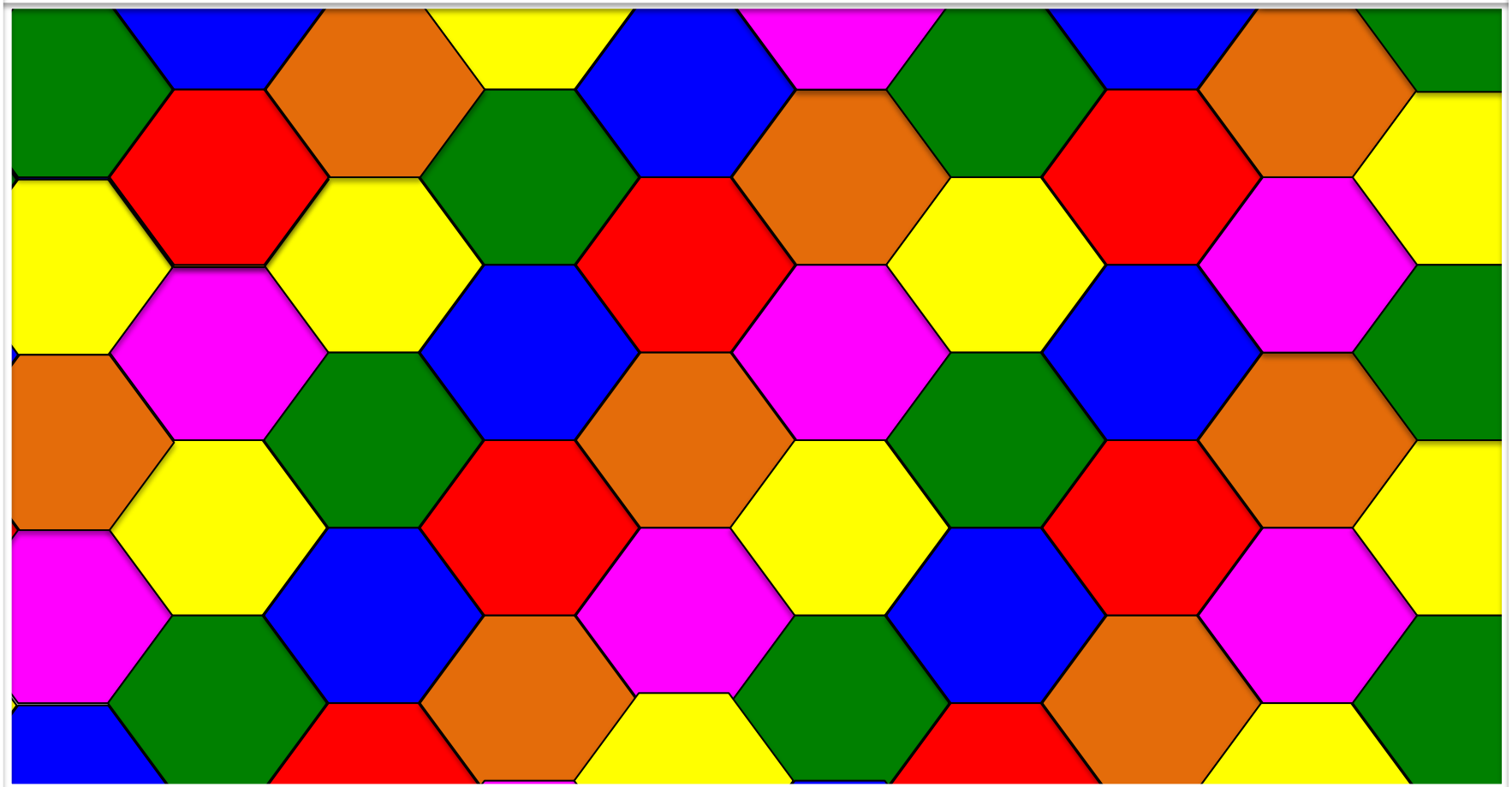
Mozaic cu piese hexagonale



Mozaic cu piese hexagonale



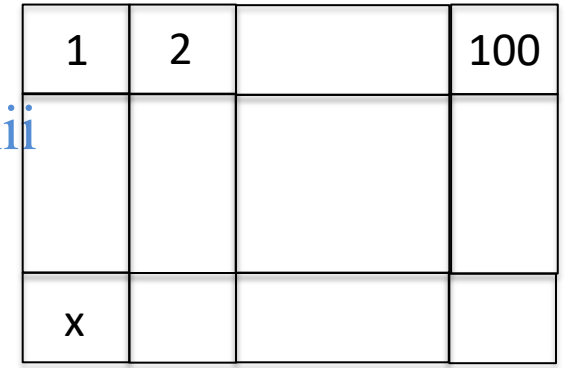
Mozaic cu piese hexagonale



Algoritm – varianta 2

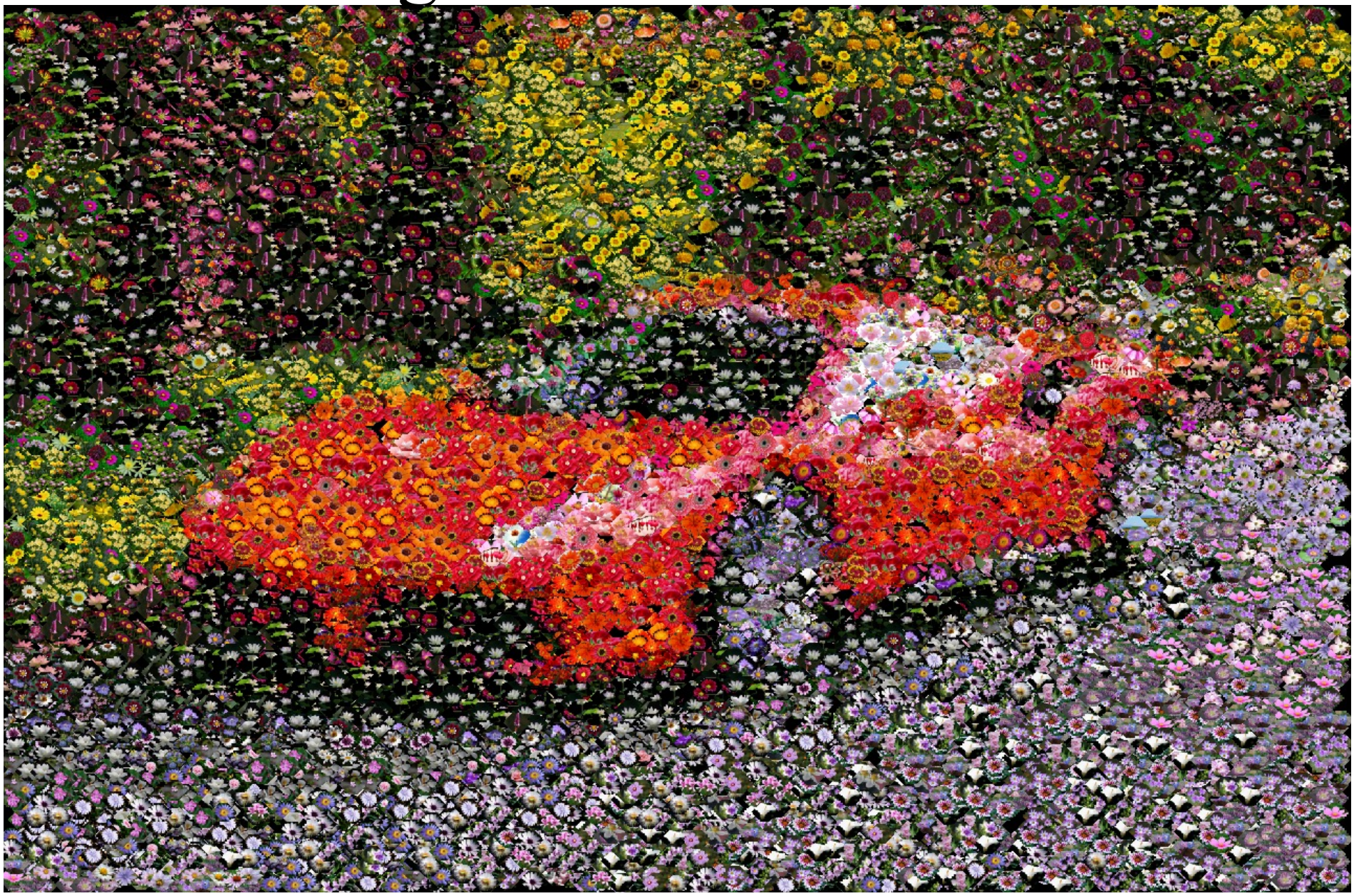
Pași:

1. determină înălțimea mozaicului:
   - păstrează proporția (aspect ratio) imaginii de referință inițiale
   - multiplu de înălțimea unei piese



2. adaugă piese în mozaic în poziții aleatoare: aplică un criteriu pentru a selecta piesa care se potrivește cel mai bine într-o pozitie:
   - aleator (alegem piesele întâmplător)
   - culoarea medie cea mai apropiată (cea mai mica distanță euclidiană)

Algoritm – varianta 2

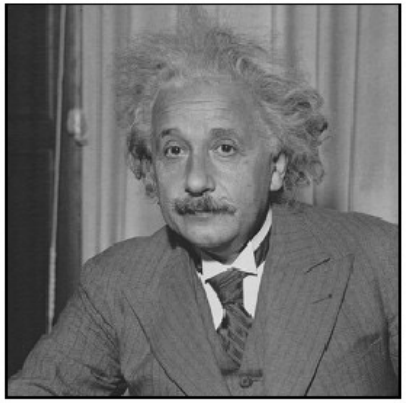


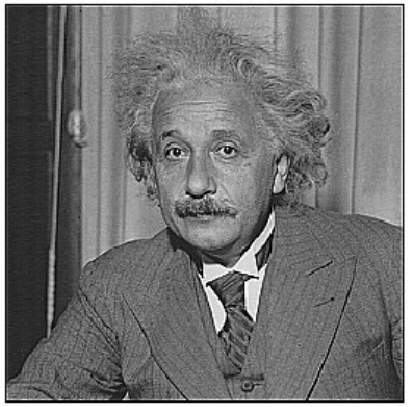
Algoritm – varianta 2

Filtrarea imaginilor

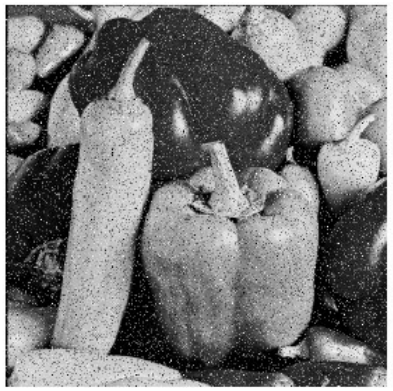
Exemple de filtrare

Filtru de blurare



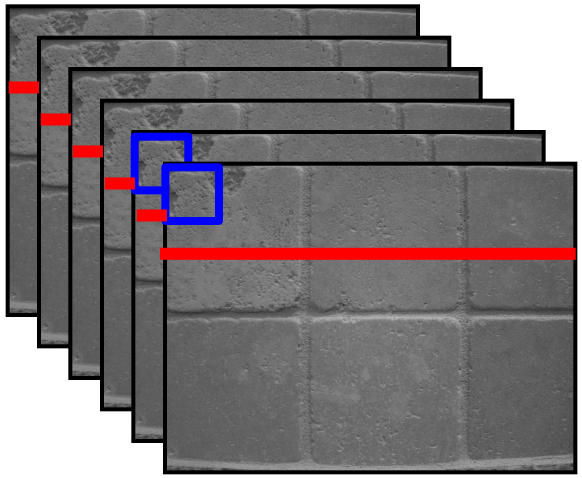


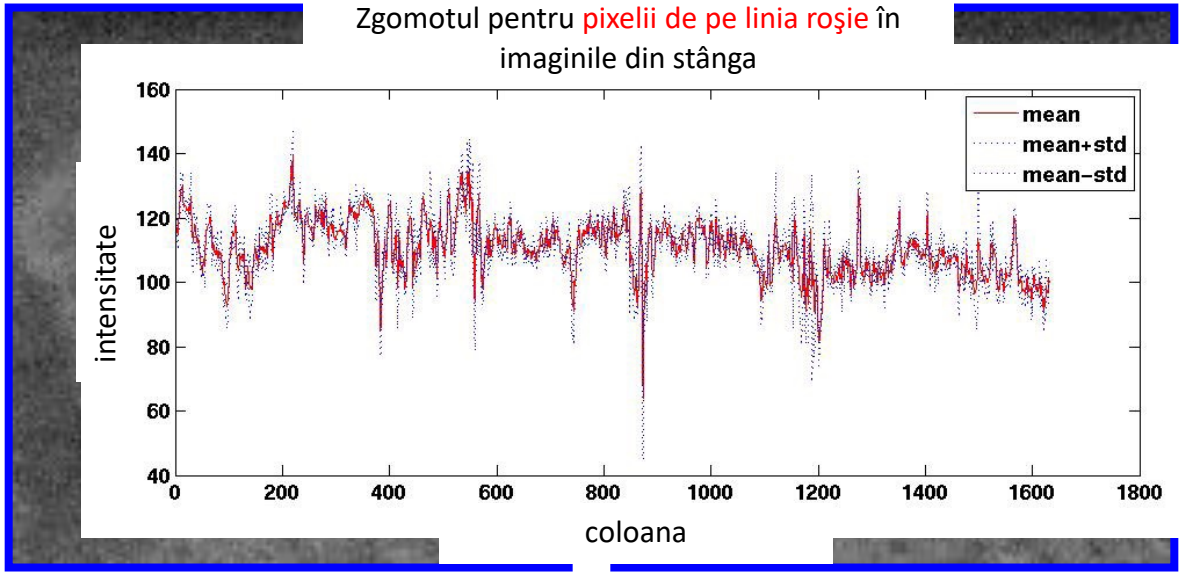
Filtru de accentuare



Filtru pentru eliminarea / reducerea zgomotului

Motivație: reducerea de zgomot





- Imperfecțiuni tehnologice ale senzorilor de imagine

$$I(x,y) = \hat{I}(x,y) + \eta(x,y)$$

imagine obținută — imagine ideală — zgomot

- Imagini multiple ale **aceleaşi scene statice** nu vor fi identice

Slide adaptat după K. Grauman

Tipuri de zgomot

imagine originală

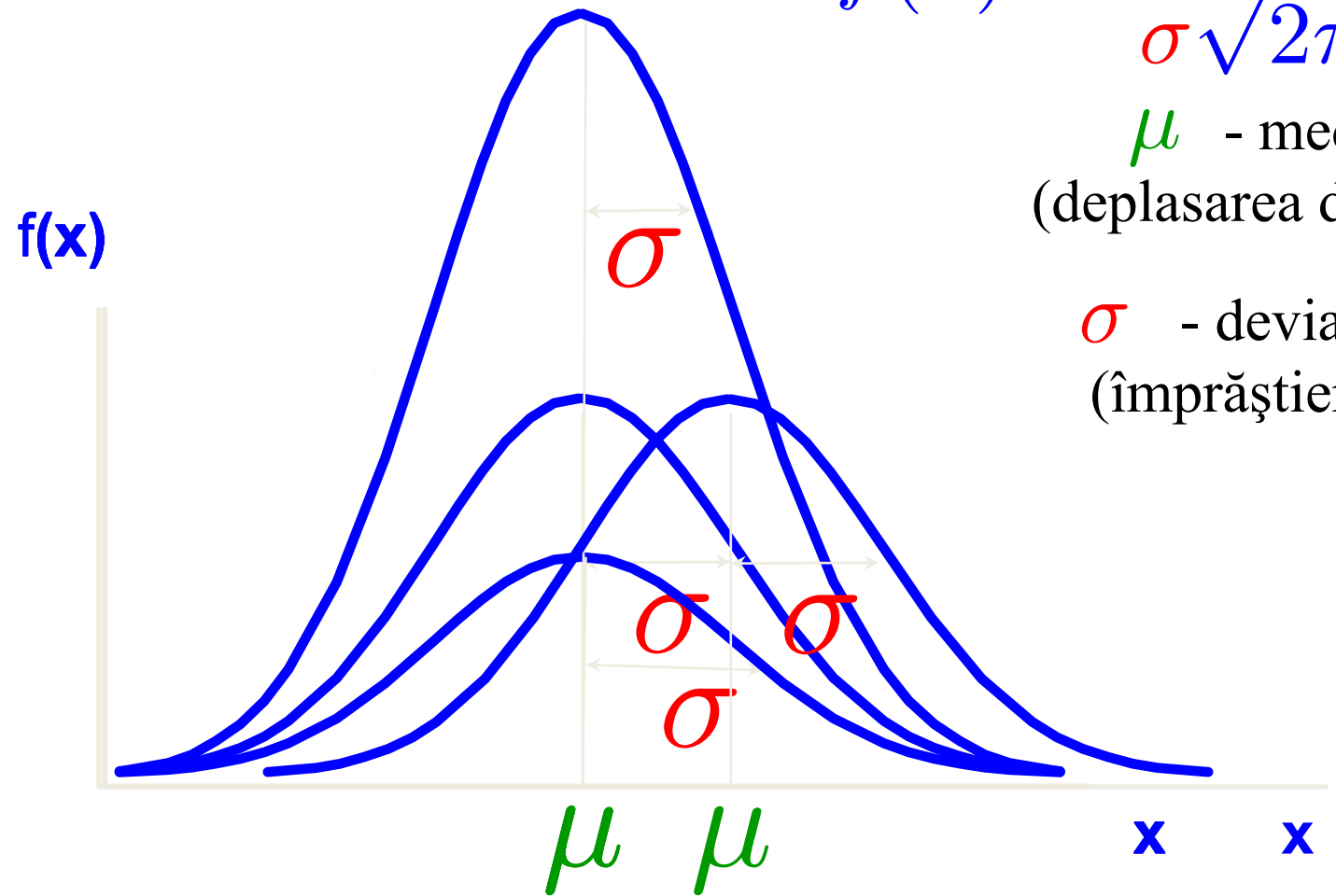
Slide adaptat după Steve Seitz

Distributia normală 1D (clopotul lui Gauss)

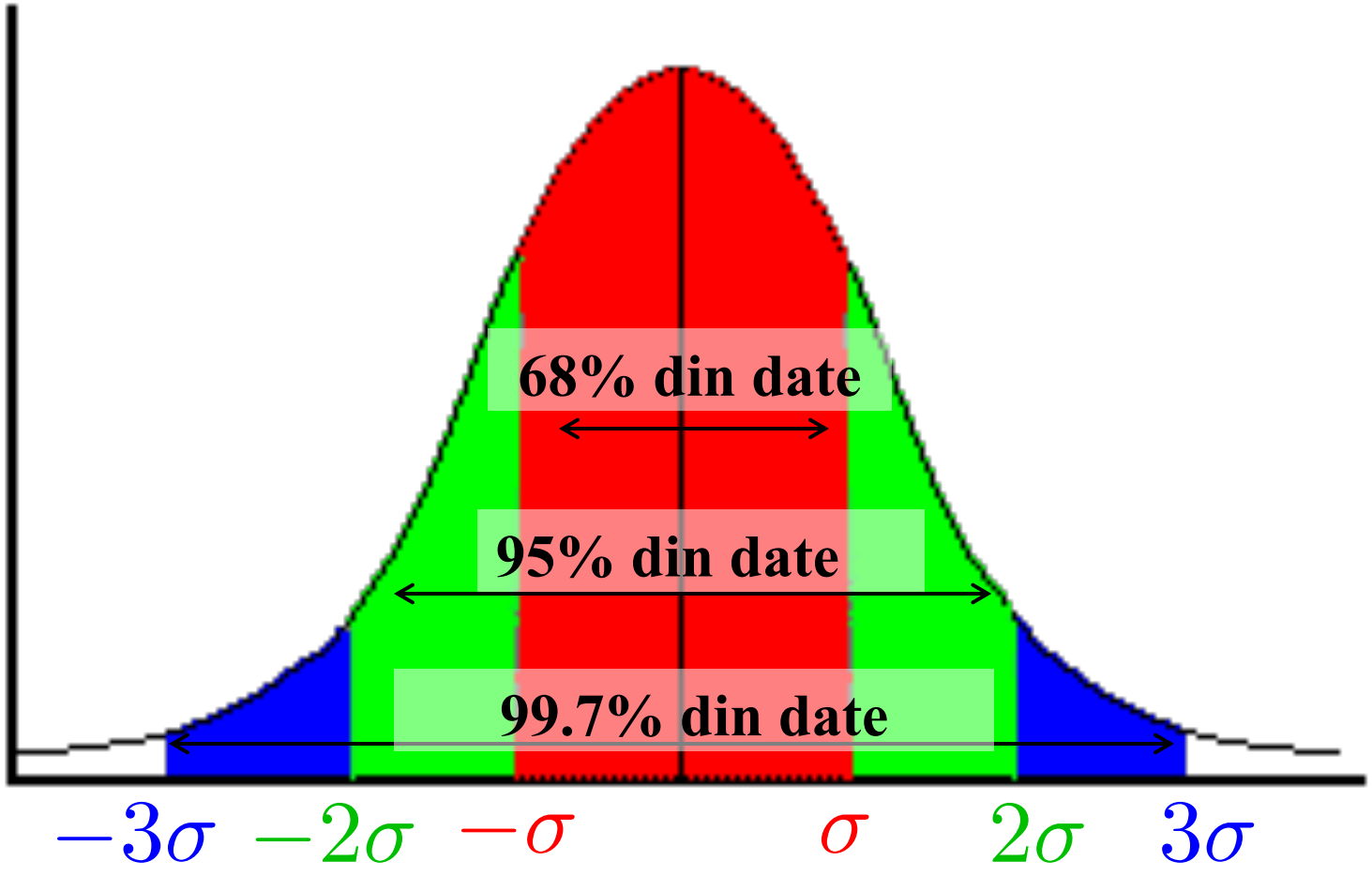
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

μ - medie
(deplasarea distribuției)

σ - deviație standard
(împrăştierea distribuției)



Regula 68-95-99.7



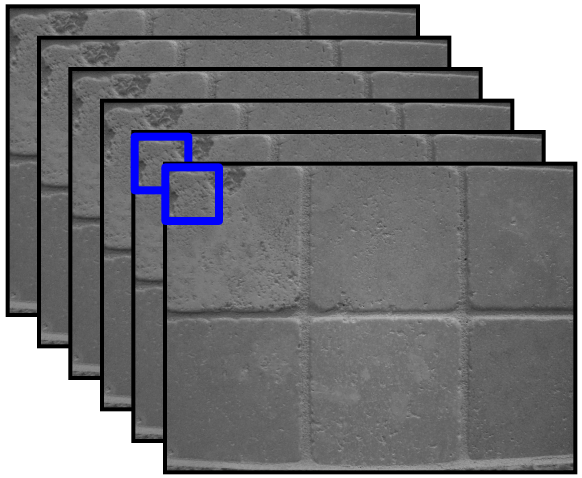
Zgomot normal – efectul lui σ

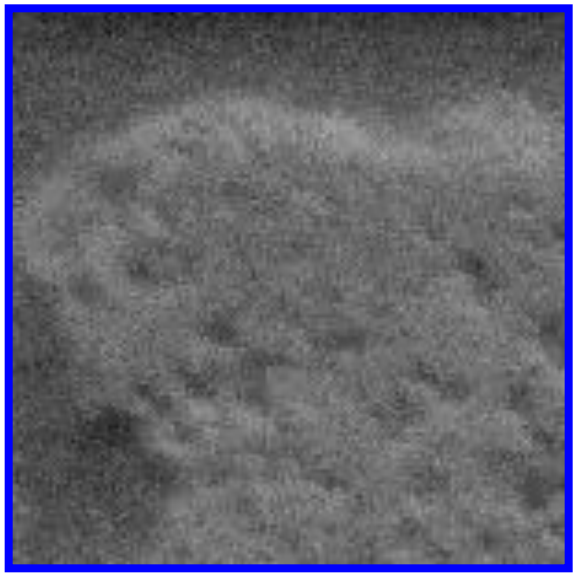
sigma = 1

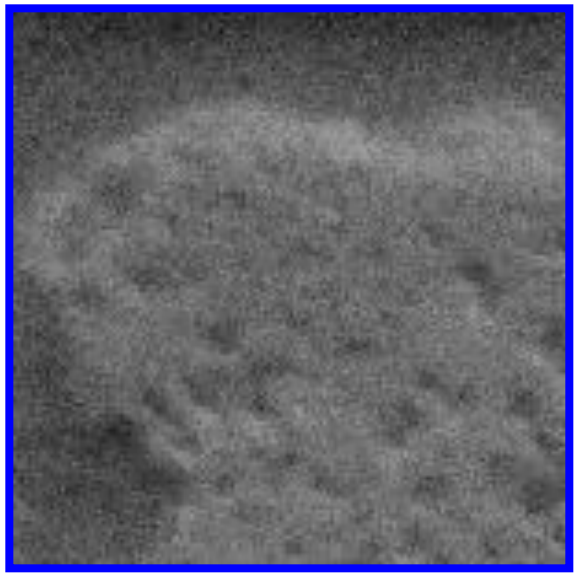
sigma = 5

sigma = 25

Motivație: reducerea de zgomot







- Chiar şi imagini multiple ale **aceleaşi scene statice** nu vor fi identice.
- Cum putem reduce zgomotul, adică să estimăm adevăratele intensități?
- Dacă avem mai multe imagini luăm media
- **Dar dacă avem numai o singură imagine?**

Slide adaptat după K. Graumann

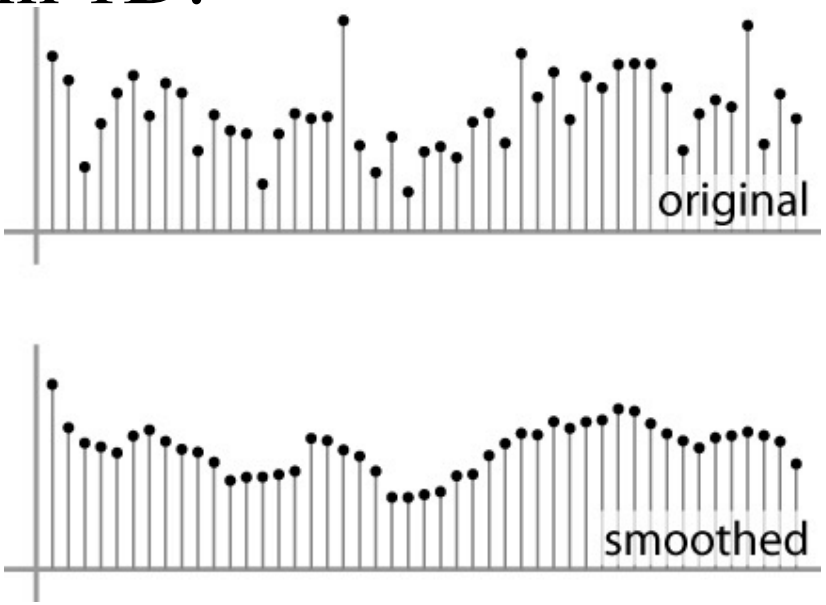
O posibilă soluție

- Ipoteze:
  - pixelii “vecini” nu diferă prea mult între ei
  - zgomotul care afectează pixelii este independent

- Soluție:
  - înlocuim valoarea fiecărui pixel cu media valorilor pixelilor din “vecinătatea” lui

O posibilă soluție

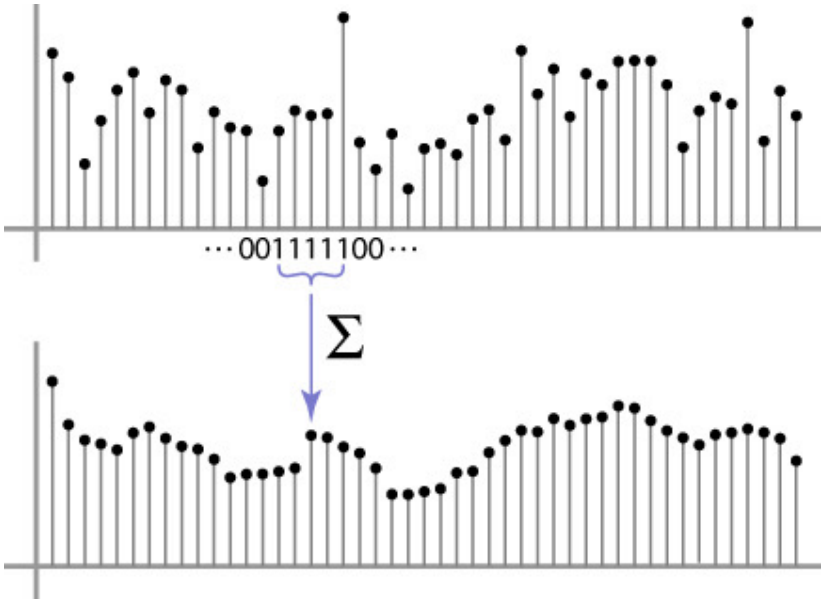
- Înlocuim valoarea fiecărui pixel cu media valorilor pixelilor din vecinătatea lui
- Exemplu în 1D:



Slide adaptat după S. Marschner

Medie ponderată

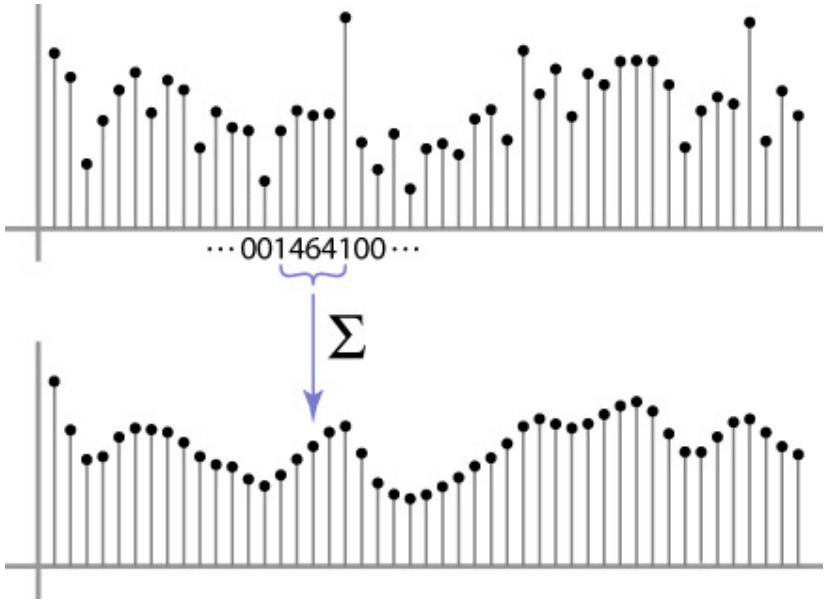
- Putem adaugă ponderi pentru fiecare pixel
- *Ponderi* [1, 1, 1, 1, 1] / 5



Slide adaptat după S. Marschner

Medie ponderată

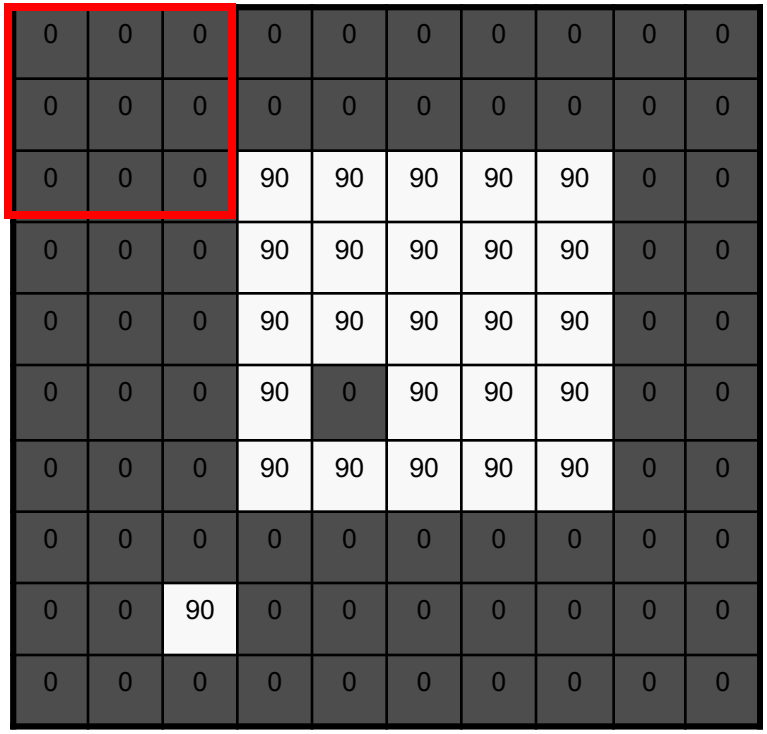
- Ponderi ne-uniforme [1, 4, 6, 4, 1] / 16



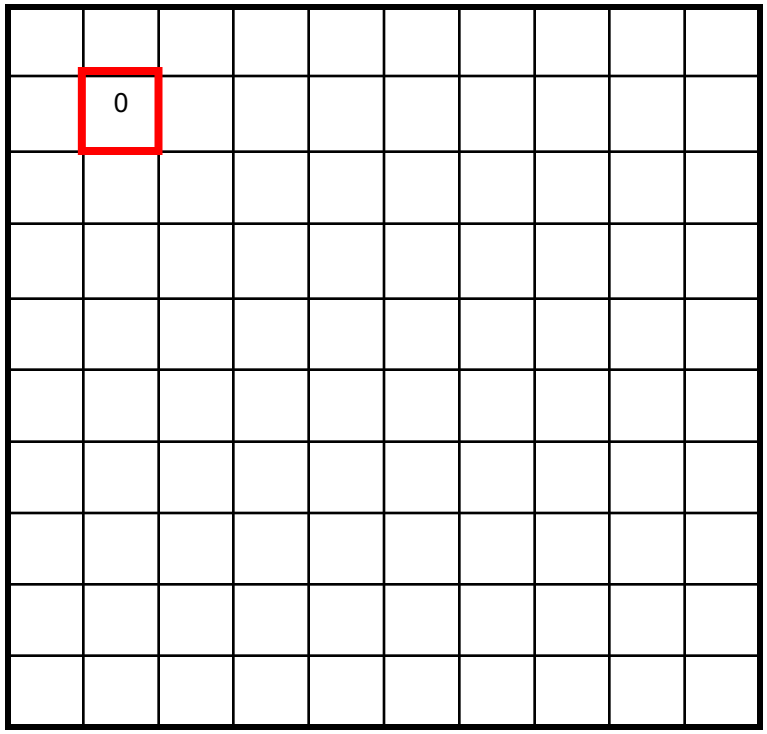
Slide adaptat după S. Marschner

Medie în 2D

$I[x, y]$



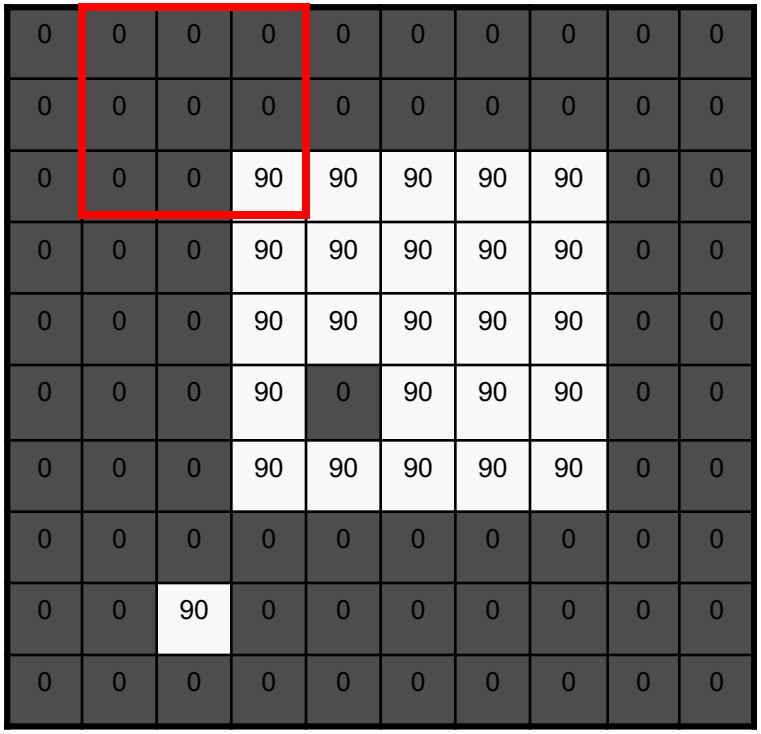
$O[x, y]$



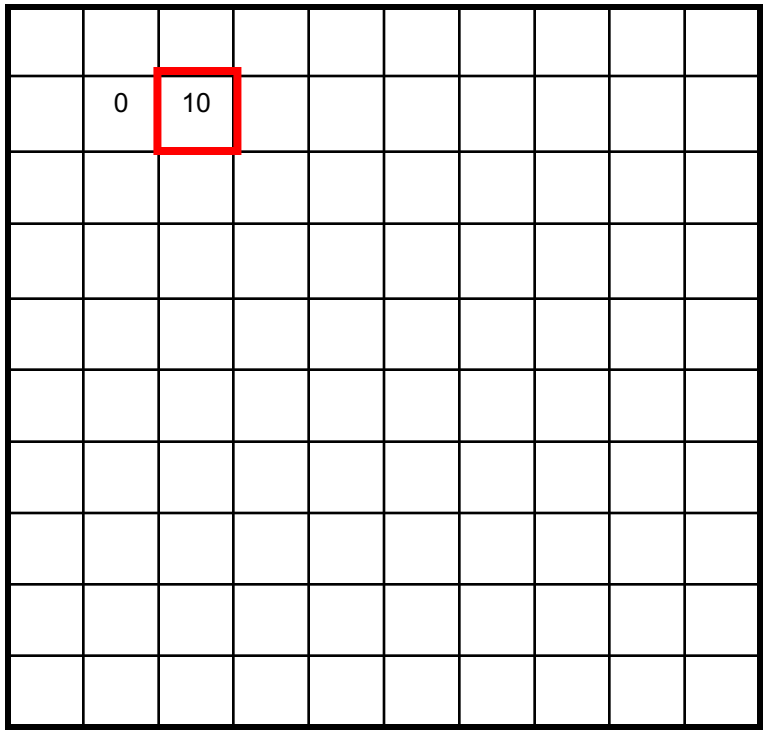
Slide adaptat după Steve Seitz

Medie în 2D

$I[x, y]$



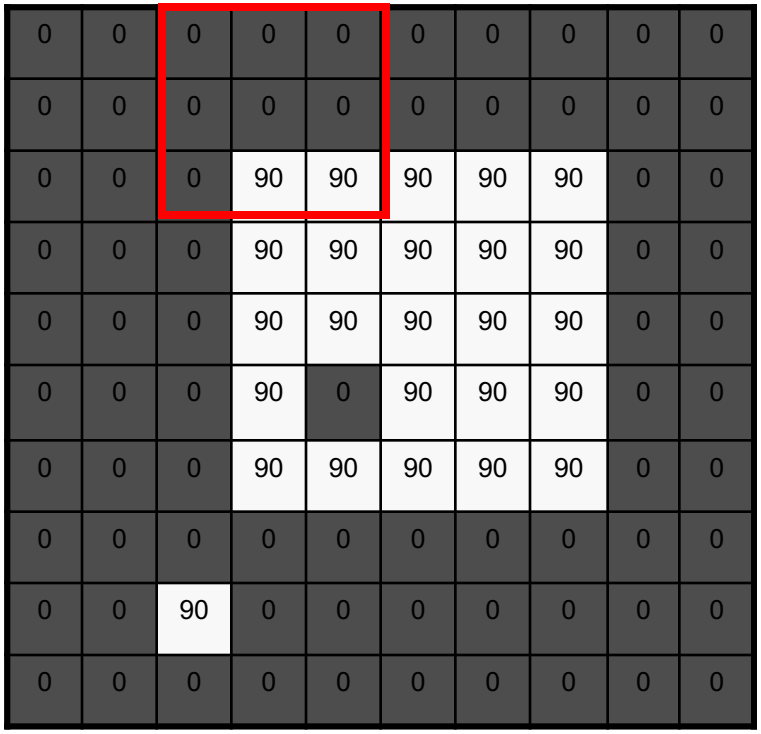
$O[x, y]$



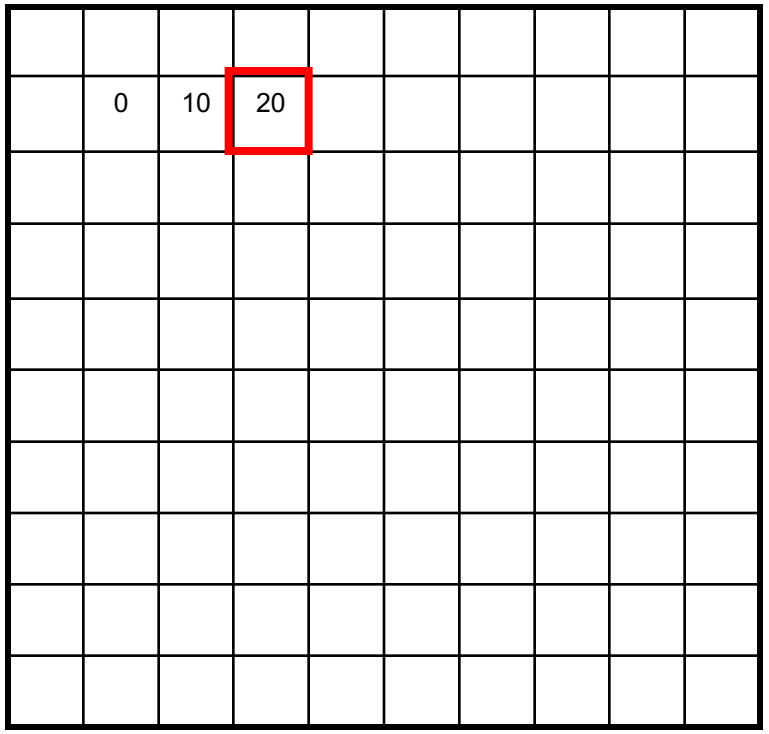
Slide adaptat după Steve Seitz

Medie în 2D

$I[x, y]$



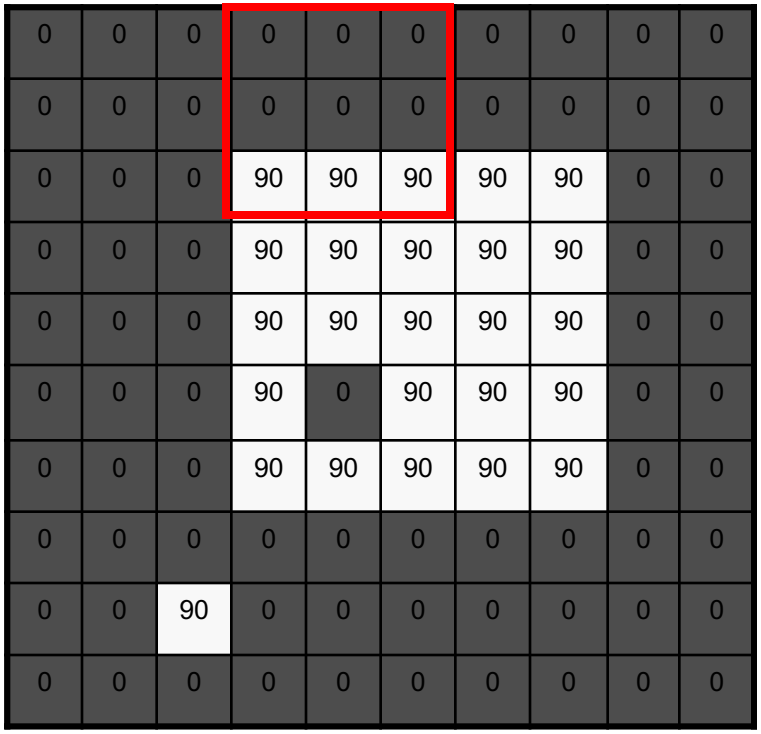
$O[x, y]$



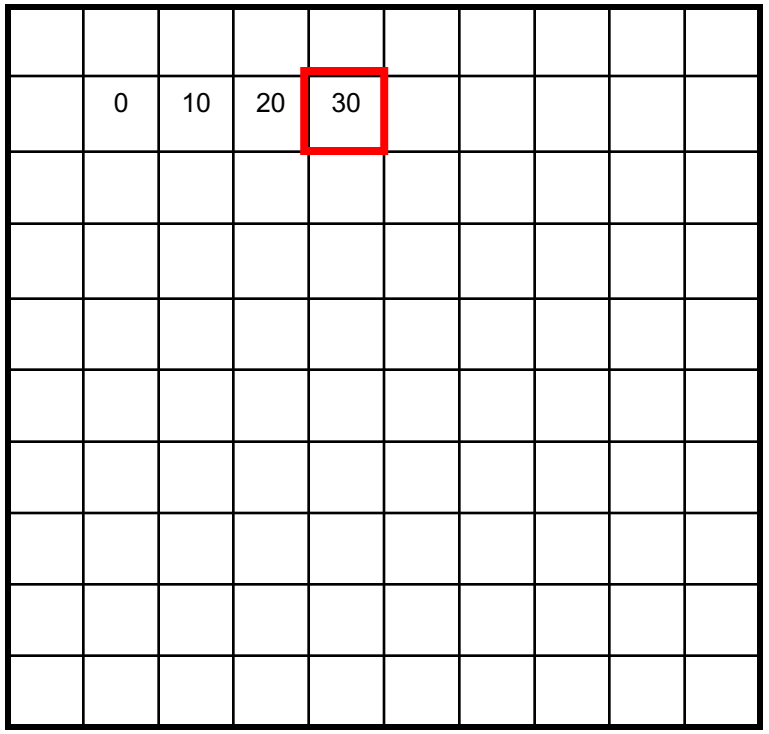
Slide adaptat după Steve Seitz

Medie în 2D

$I[x, y]$



$O[x, y]$



Slide adaptat după Steve Seitz

Medie în 2D

$I[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$O[x, y]$

	0	10	20	30	30				

Slide adaptat după Steve Seitz

Medie în 2D

$I[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$O[x, y]$

	0	10	20	30	30				
							?		

Slide adaptat după Steve Seitz

Medie în 2D

$I[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$O[x, y]$

	0	10	20	30	30				
							60		

Slide adaptat după Steve Seitz

Medie în 2D

$I[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$O[x, y]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Slide adaptat după Steve Seitz

Corelație

- Pentru o vecinătate de dimensiuni 2k+1 × 2k+1:

$$O[i,j] = \underbrace{\frac{1}{(2k+1)^2}}_{\text{pondere egală pentru fiecare pixel}} \underbrace{\sum_{u=-k}^{k} \sum_{v=-k}^{k} I[i+u, j+v]}_{\text{toți pixelii din vecinătatea pixelului (i,j) din imaginea I}}$$

(i-k,j-k)				(i-k,j+k)
		(i,j)		
(i+k,j-k)				(i+k,j+k)

Vecinătate de dimensiuni (2k +1) x (2k+1) centrată în (i,j)

Slide adaptat după K. Grauman

Corelație

- Pentru o vecinătate de dimensiuni 2k+1 × 2k+1:

$$O[i,j] = \underbrace{\frac{1}{(2k+1)^2}}_{\text{pondere egală pentru fiecare pixel}} \underbrace{\sum_{u=-k}^{k}\sum_{v=-k}^{k} I[i+u, j+v]}_{\text{toți pixelii din vecinătatea pixelului (i,j) din imaginea I}}$$

- Generalizăm pentru a permite ponderi diferite:

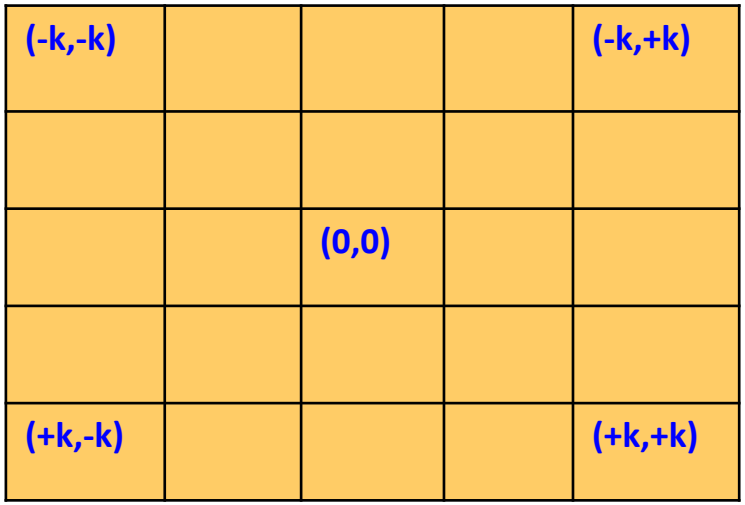
$$O[i,j] = \sum_{u=-k}^{k}\sum_{v=-k}^{k} \underbrace{F[u,v]}_{\text{ponderi ne-uniforme}} I[i+u, j+v]$$

Slide adaptat după K. Grauman

Corelație

$$O[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[u,v] I[i+u, j+v]$$

Corelație: $O = F \otimes I$



Filtrul F de dimensiuni (2k +1) x (2k+1) centrat în (0,0)

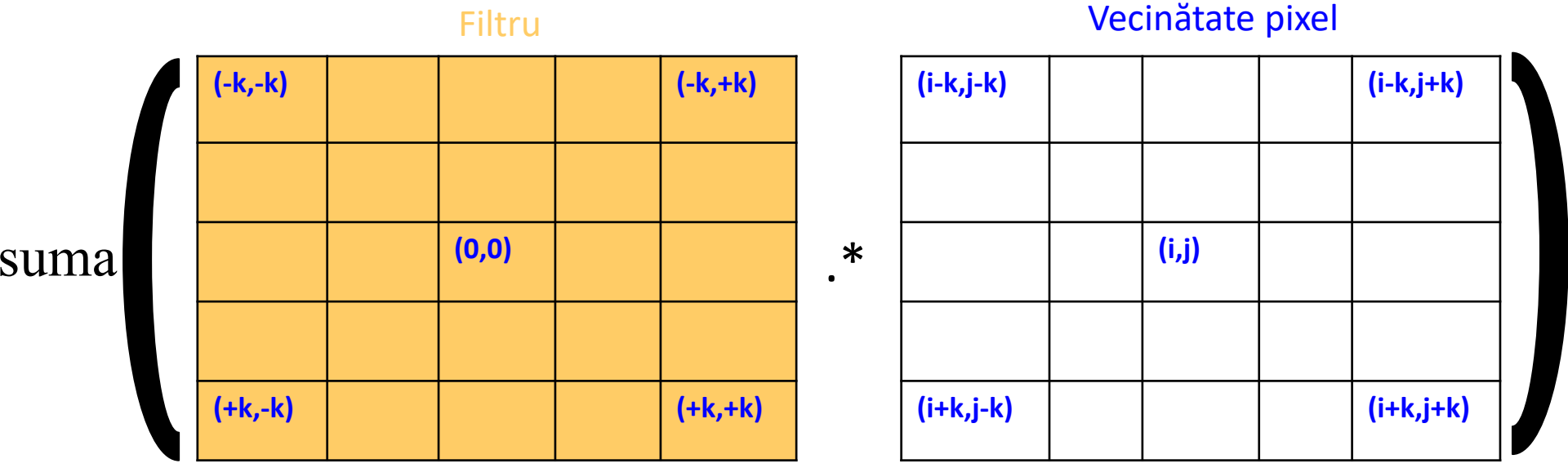
Slide adaptat după K. Grauman

Corelație

$$O[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[u,v] I[i+u, j+v]$$

Corelație: $O = F \otimes I$

Filtrarea (liniară a) unei imagini: înlocuim fiecare pixel cu o combinație (liniară) a “vecinilor” săi.



Corelație

$$O[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[u,v] I[i+u, j+v]$$

Corelație: $O = F \otimes I$

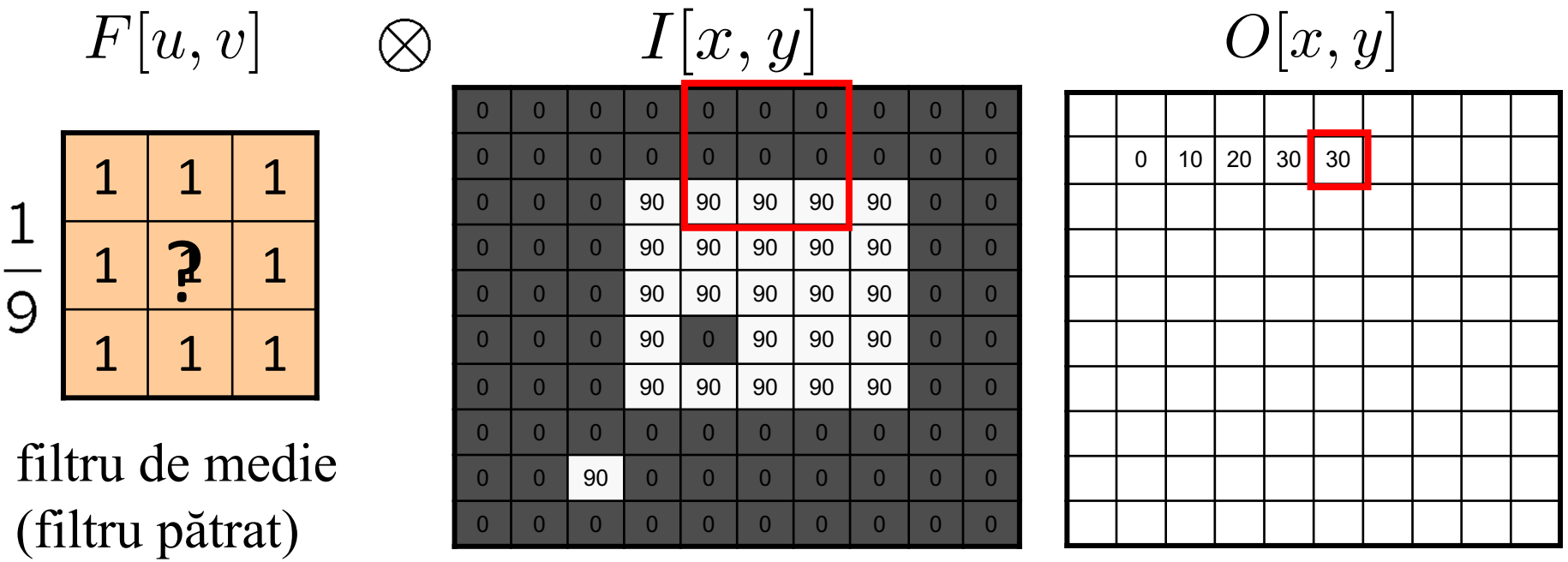
Filtrarea (liniară a) unei imagini: înlocuim fiecare pixel cu o combinație (liniară) a "vecinilor" săi.

F se mai numeşte filtru, kernel, mască (de filtrare).

Filtrul F - conține ponderile pixelilor folosiți în combinația (liniară).

Filtrul de medie

- Ce ponderi are filtrul *F* pentru exemplul precedent (medie în 2D)?

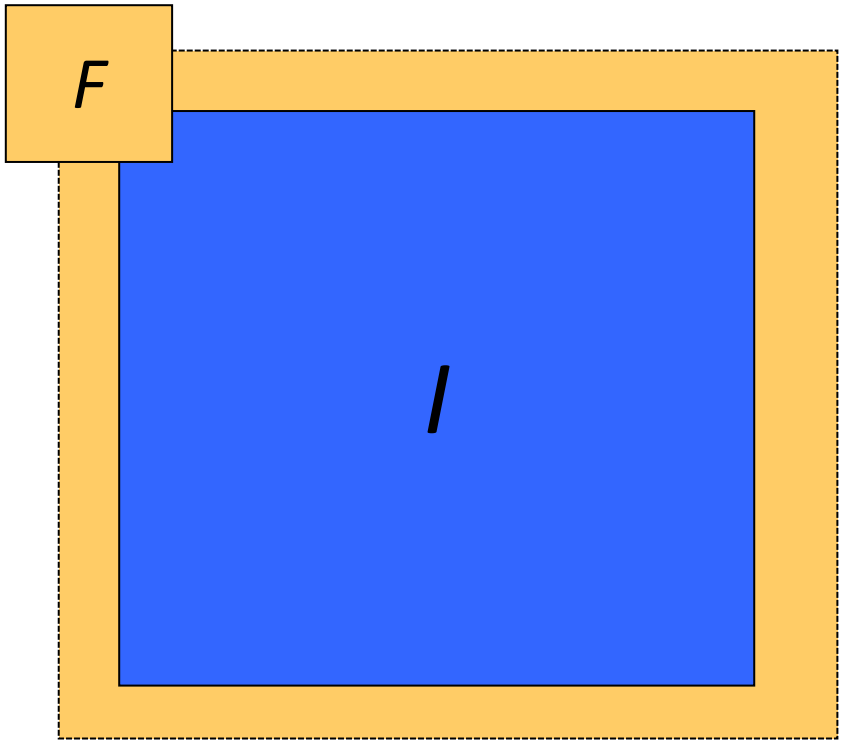


$$O = F \otimes I$$

Slide adaptat după Steve Seitz

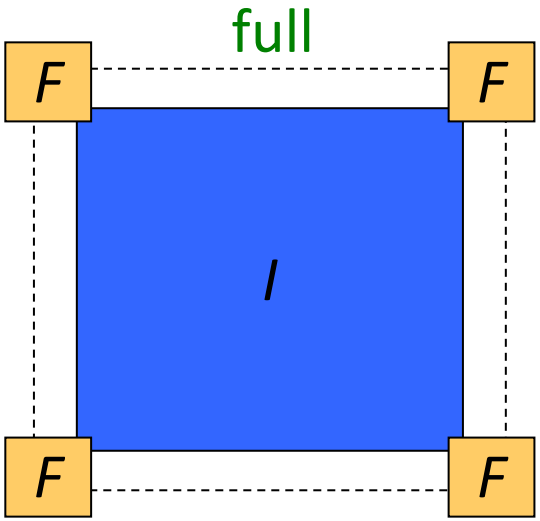
Dimensiune output

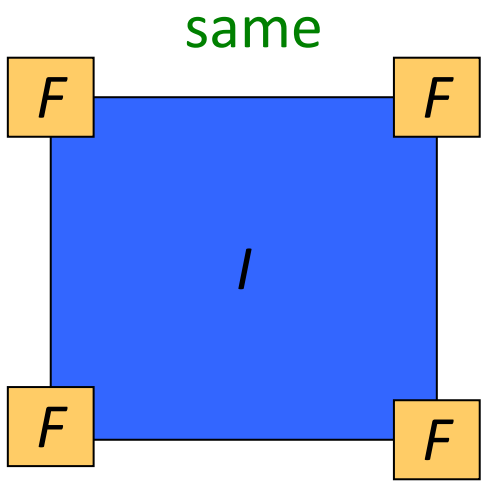
- Care este dimensiunea imaginii filtrate?

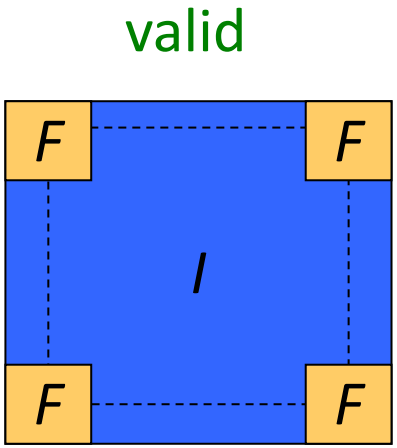


Dimensiune output

- Care este dimensiunea imaginii filtrate?
- PYTHON: scipy.signal.convolve2d
- opțiuni pentru filtrare
  - 'full': dim(O) = dim(I) + dim(F) - 1
  - 'same': dim(O) = dim(I)
  - 'valid': dim(O) = dim(I) – dim(F) + 1



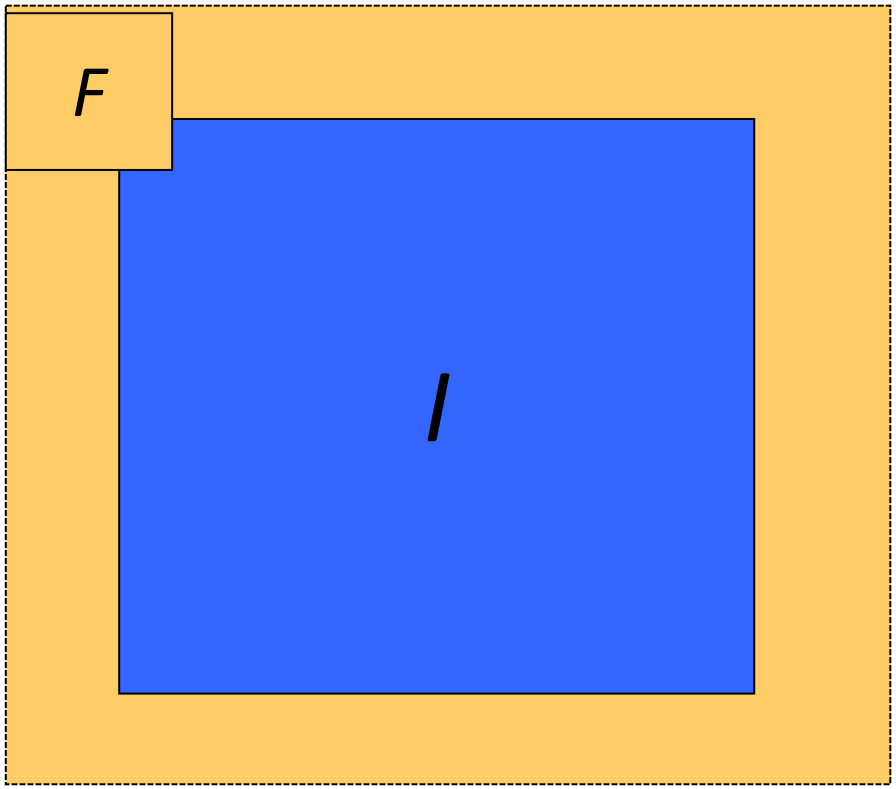




Slide adaptat după S. Lazebnik

Pixelii din afara imaginii

- Ce valori au pixelii din afara imaginii ?



- Fereastra filtrului depăseste marginea imaginii

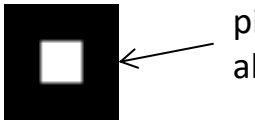
Slide adaptat după S. Marschner

Pixelii din afara imaginii

- Ce valori au pixelii din afara imaginii ?
  - extrapolăm/halucinăm valori
  1. pixeli = 0 (negru)
  2. copiază pixelii în mod circular
  3. copiază pixelii de la marginea imaginii
  4. oglindeşte pixelii de la marginea imaginii

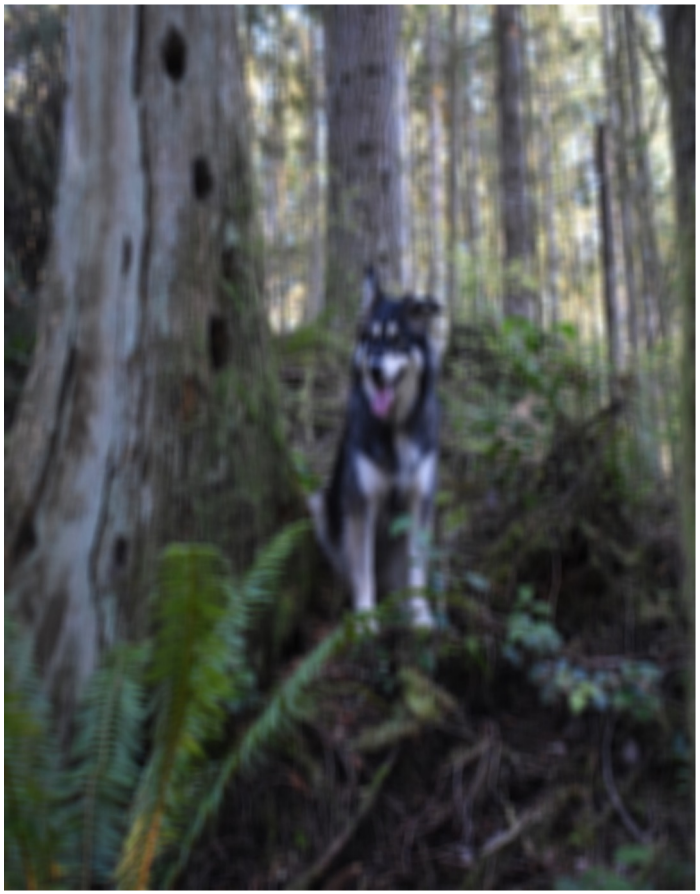
Slide adaptat după S. Marschner

Blurarea (netezirea) unei imagini cu un filtru de medie



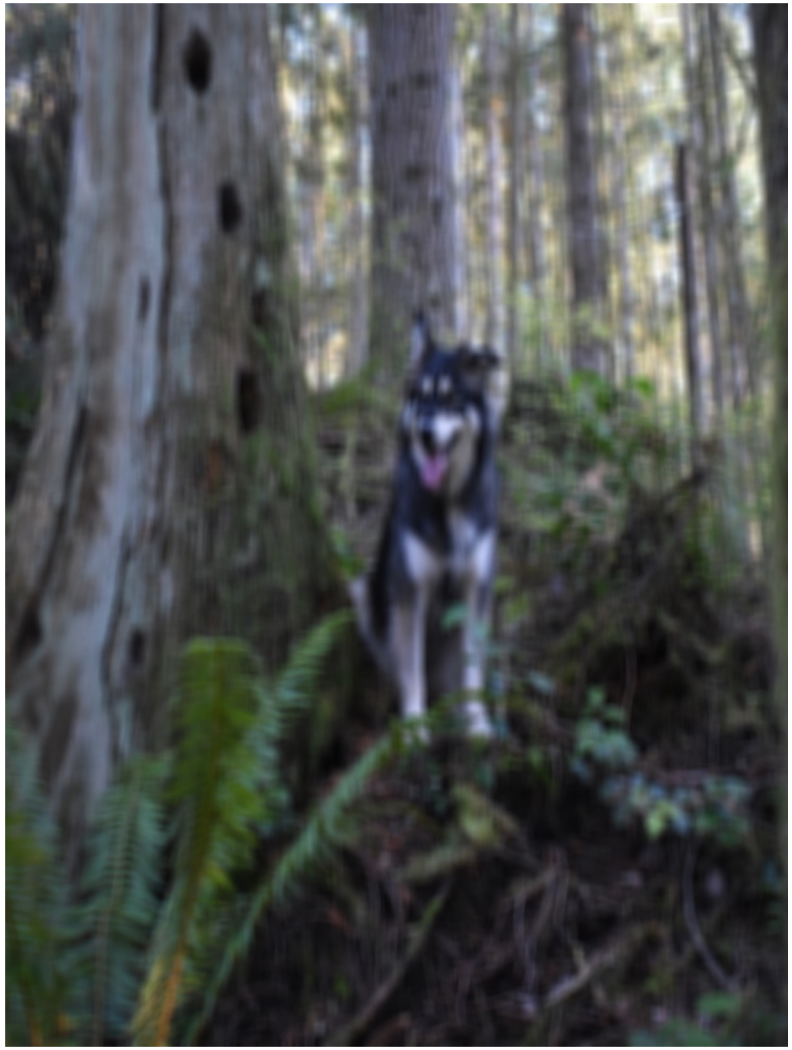
pictograma pentru filtru de medie:
alb = valoare mare, negru = valoare mică

Imagine inițială

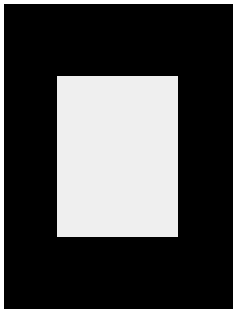


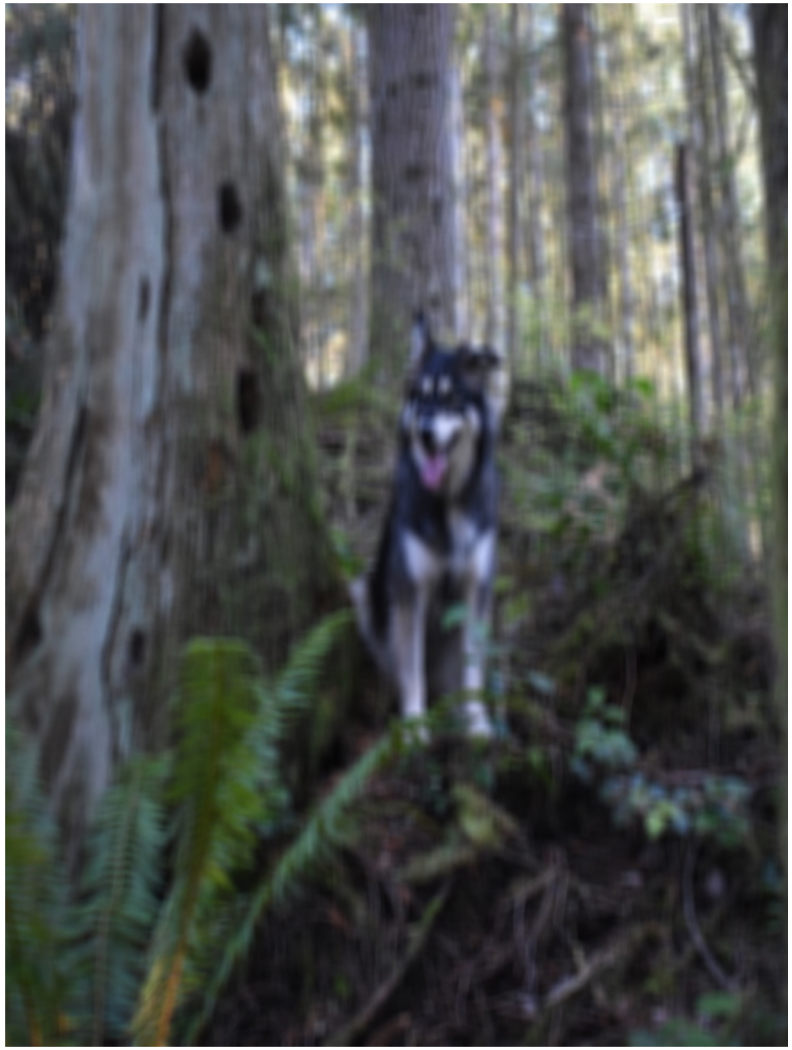
Imagine filtrată

Filtrul de medie prezintă efecte secundare

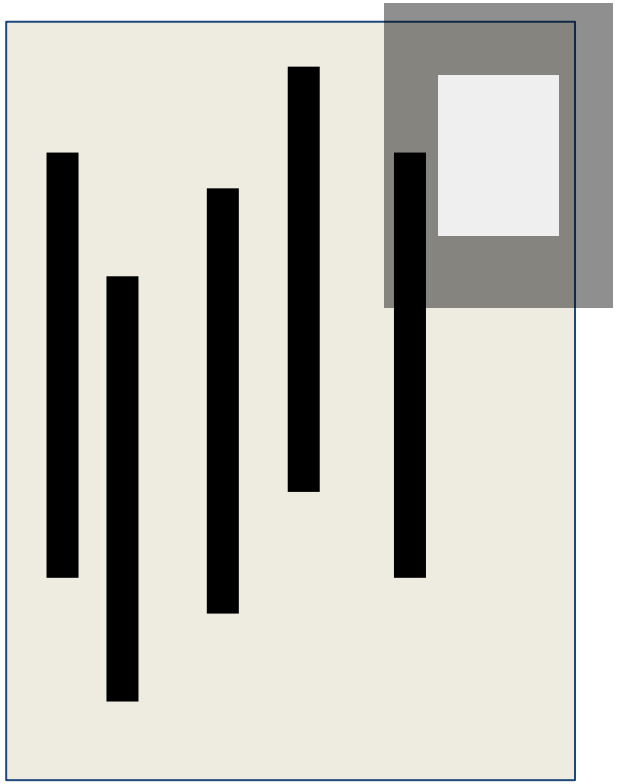


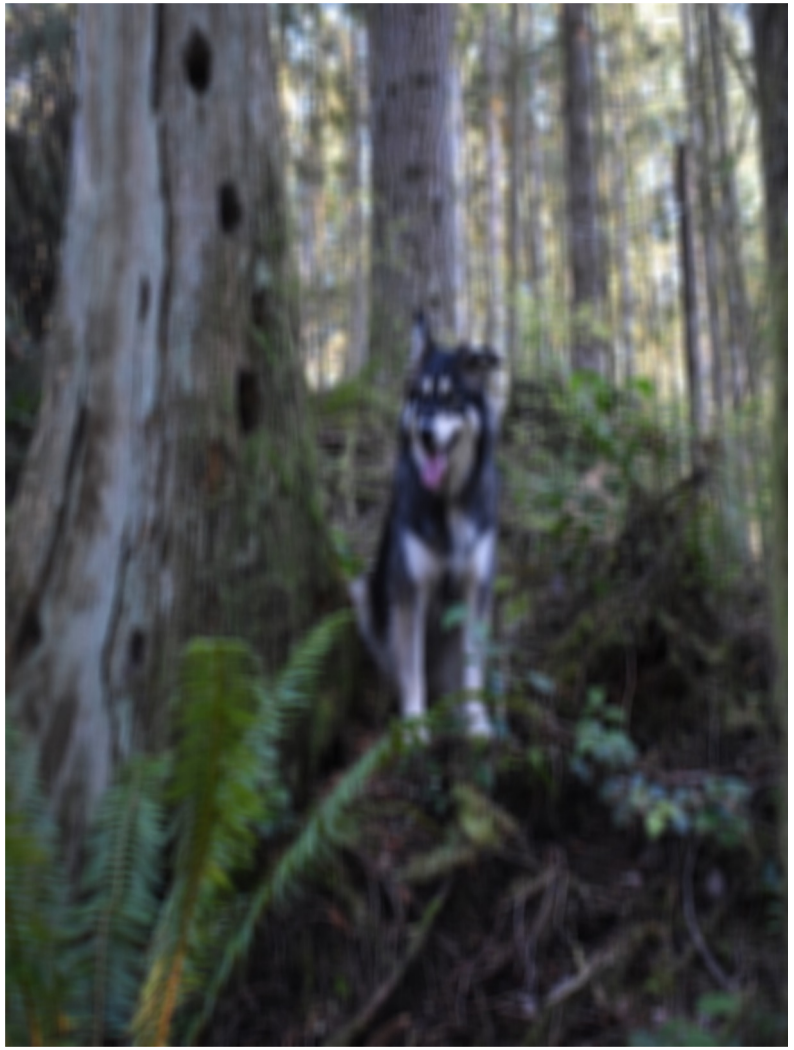
Dungi verticale și orizontale



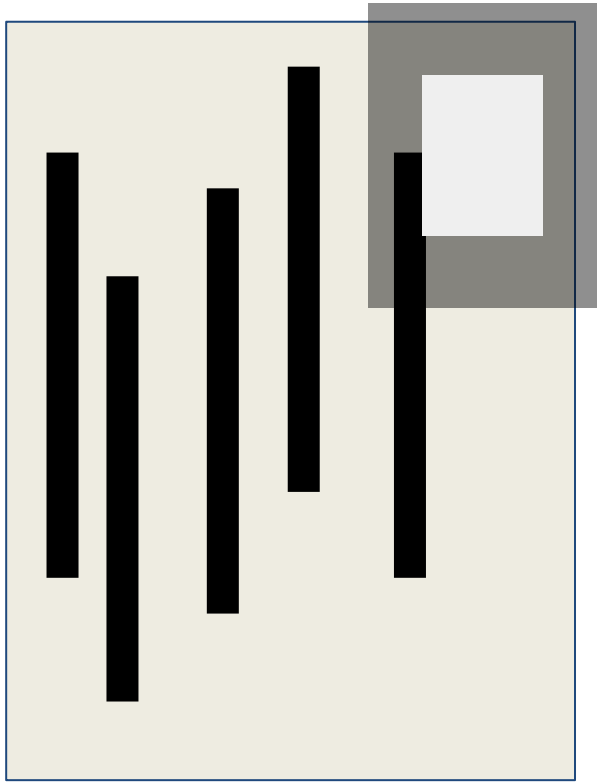


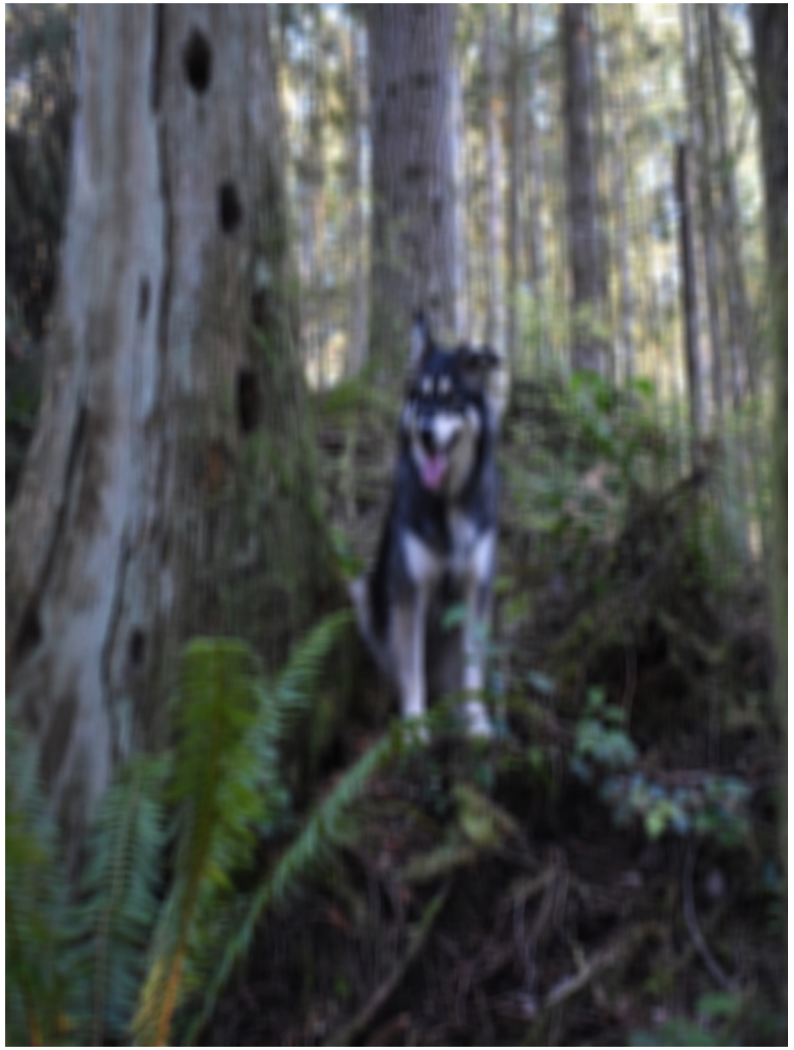
Dungi verticale și orizontale



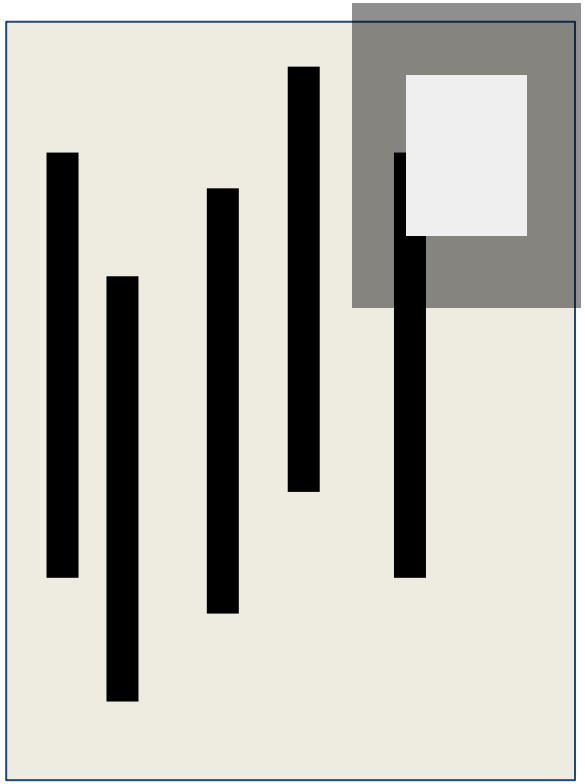


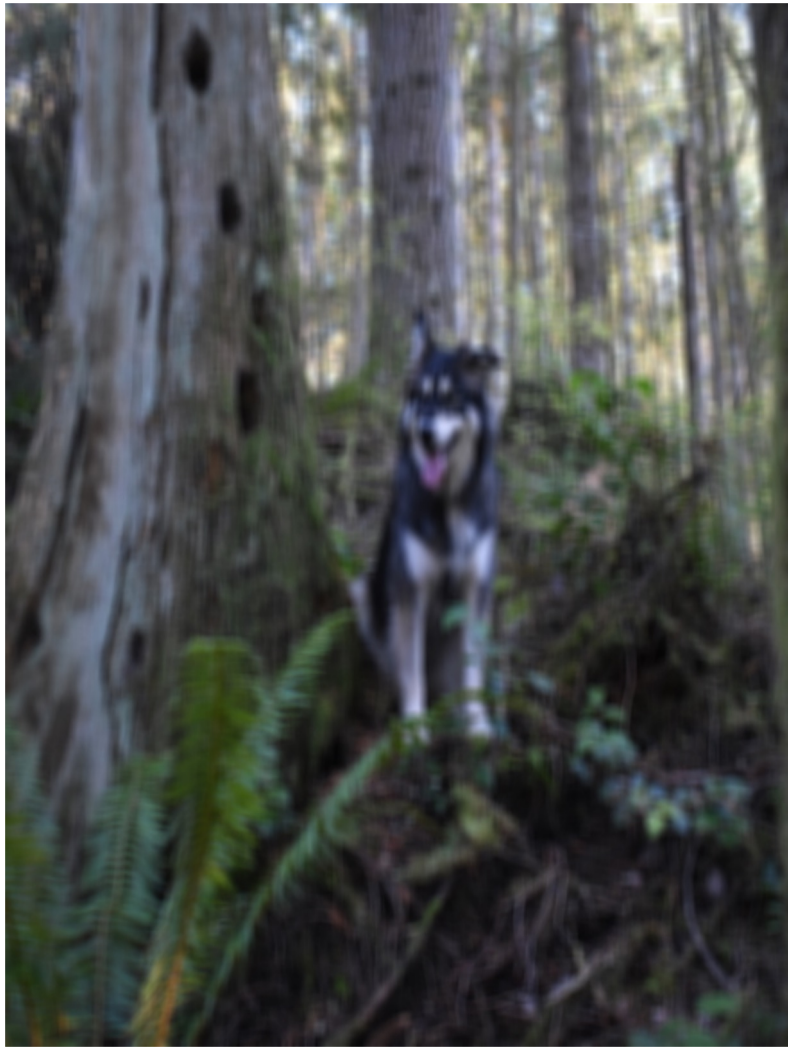
Dungi verticale și orizontale



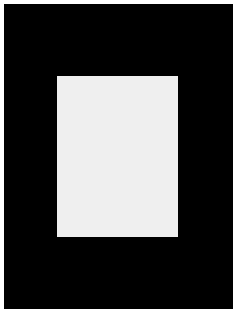


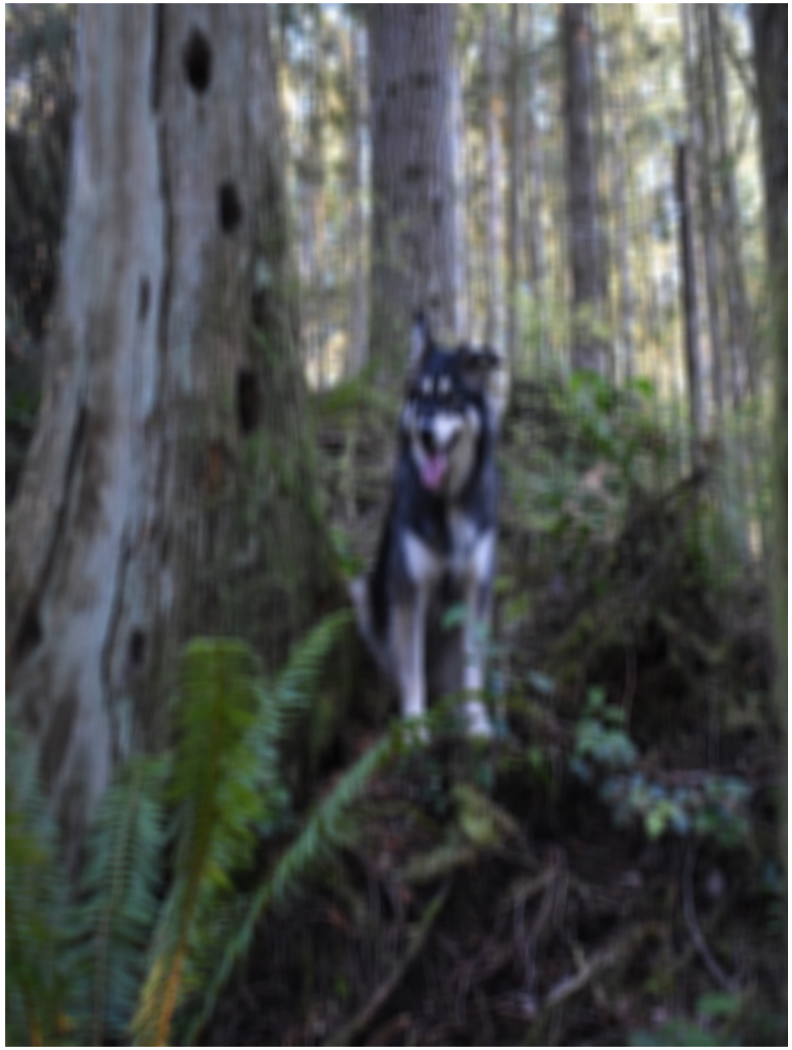
Dungi verticale și orizontale





Există alt filtru mai bun?



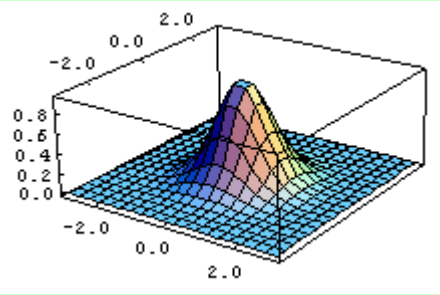


Filtru normal (Gaussian)

- Cei mai apropiați pixeli vecini au o pondere mai mare în imaginea filtrată

Acest filtru aproximează distribuția normală în 2D:

$$f(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



$$\frac{1}{16}$$

1	2	1
2	4	2
1	2	1

$F[u,v]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$I[x,y]$

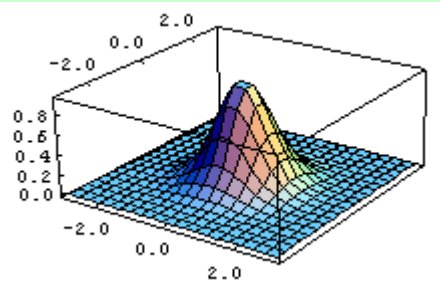
Slide adaptat după Steve Seitz

Filtru normal (Gaussian)

- Cei mai apropiați pixeli vecini au o pondere mai mare în imaginea filtrată

Acest filtru aproximează distribuția normală în 2D:

$$f(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



$$\frac{1}{16}$$

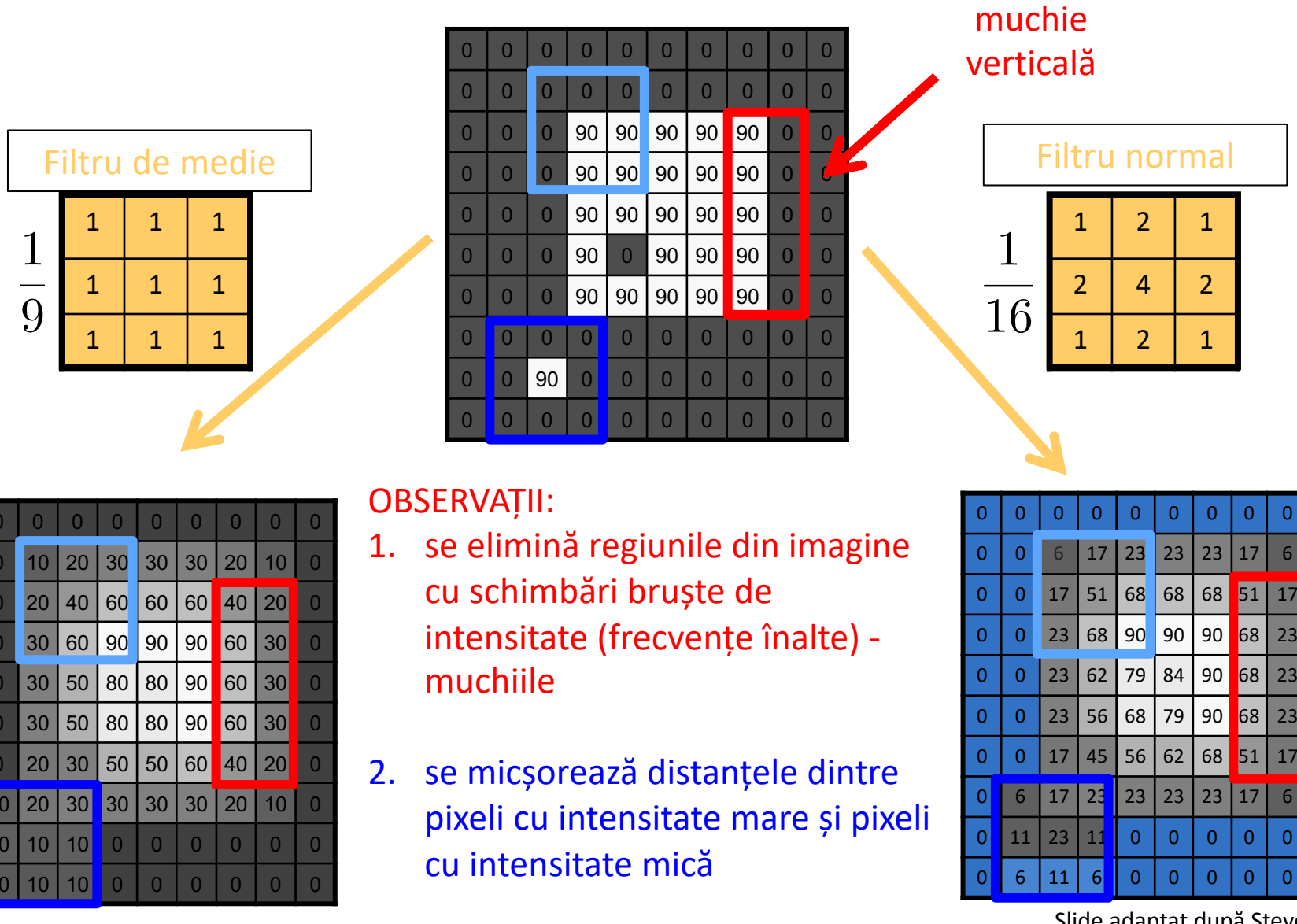
1	2	1
2	4	2
1	2	1

$F[u,v]$

0	0	0	0	0	0	0	0	0	0
0	0	6	17	23	23	23	17	6	0
0	0	17	51	68	68	68	51	17	0
0	0	23	68	90	90	90	68	23	0
0	0	23	62	79	84	90	68	23	0
0	0	23	56	68	79	90	68	23	0
0	0	17	45	56	62	68	51	17	0
0	6	17	23	23	23	23	17	6	0
0	11	23	11	0	0	0	0	0	0
0	6	11	6	0	0	0	0	0	0

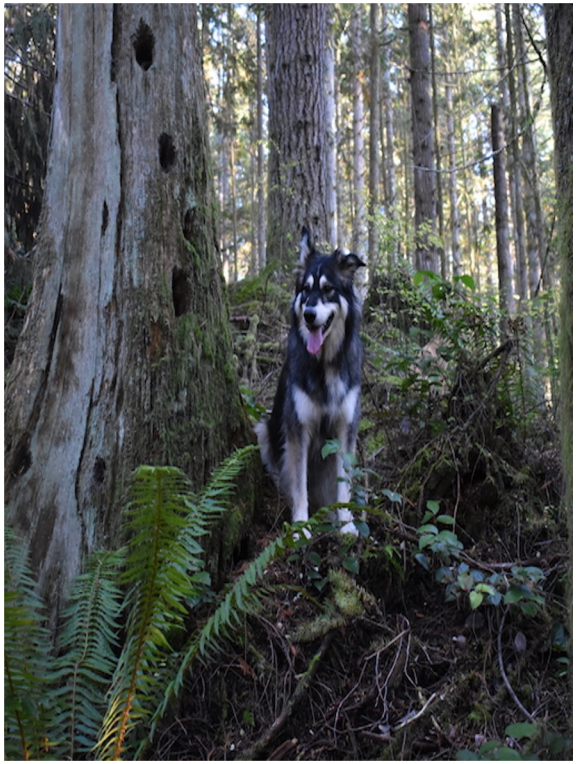
$I[x,y]$

Blurarea unei imagini

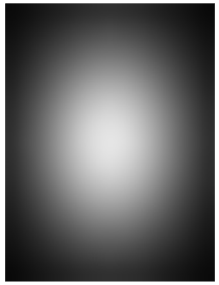


Slide adaptat după Steve Seitz

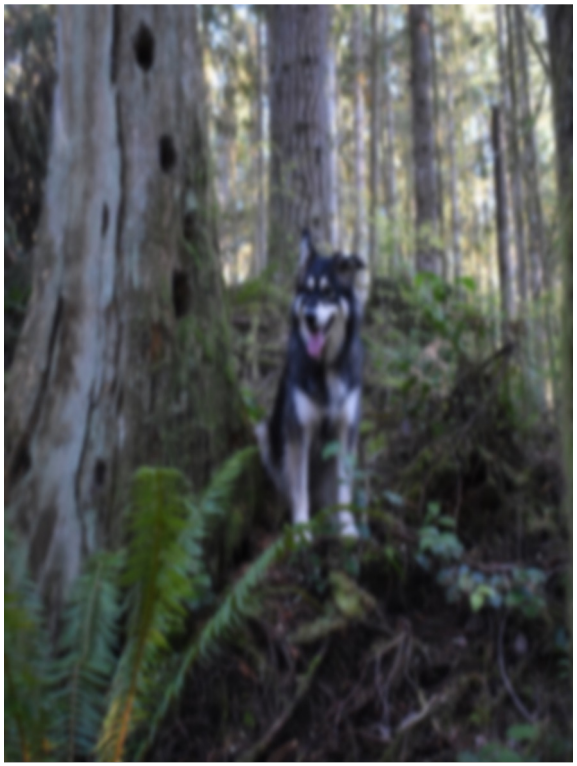
Blurarea unei imagini cu filtru Gaussian



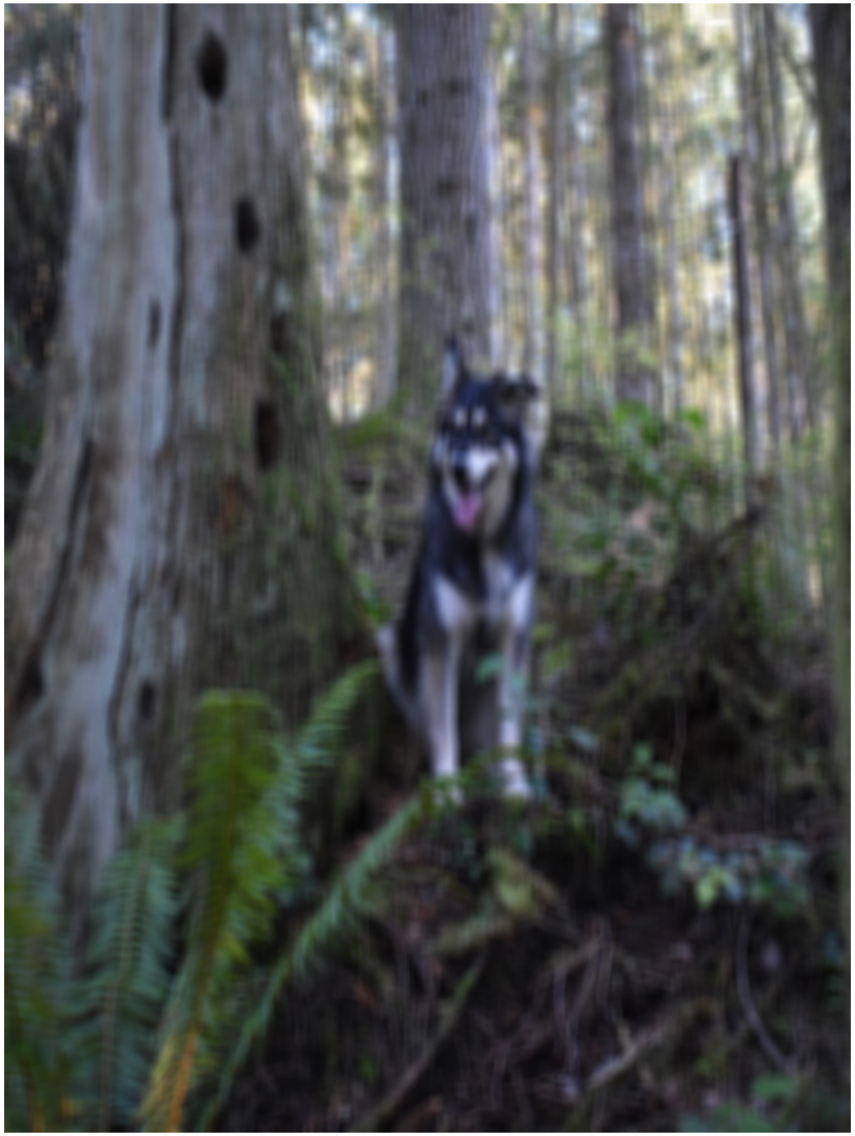
*



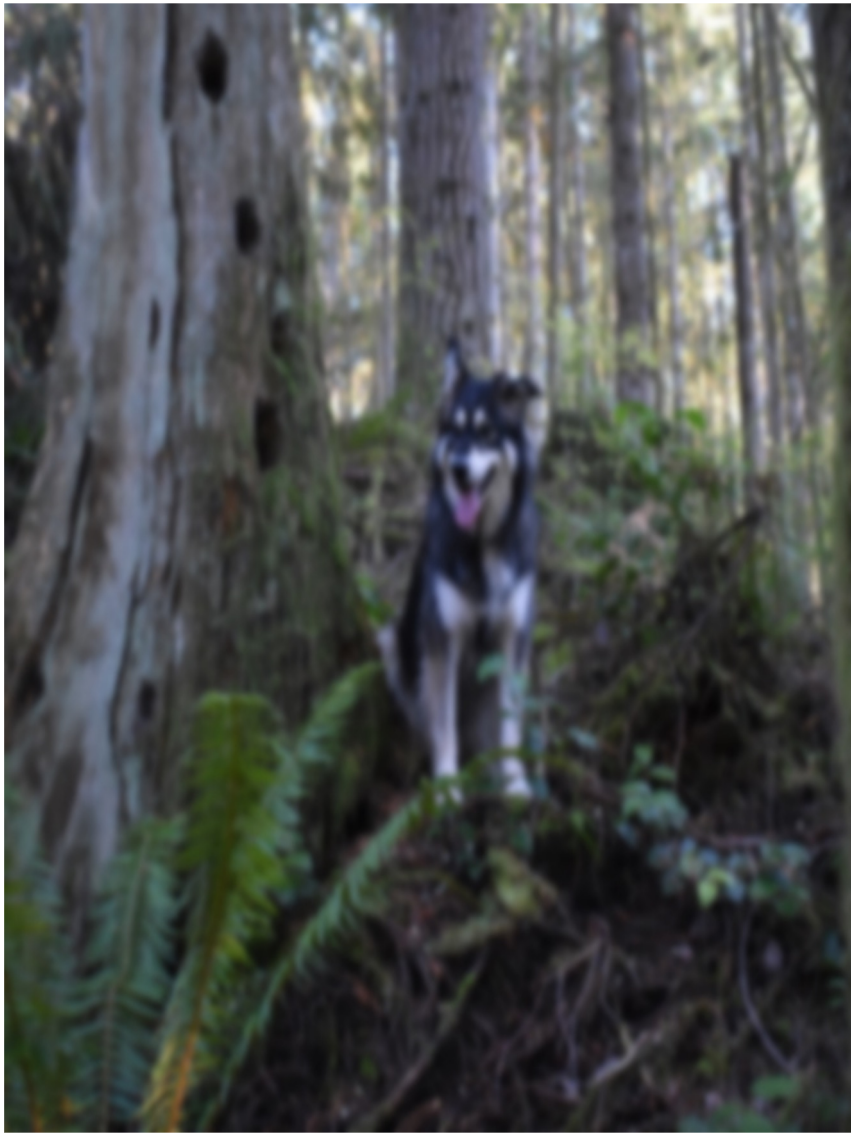
=



Blurarea unei imagini cu filtru Gaussian



filtrare cu filtru de medie



filtrare cu filtru Gaussian

Blurarea unei imagini cu filtru Gaussian

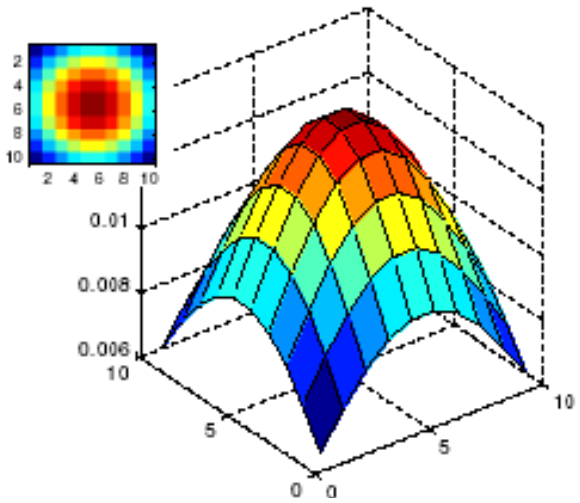
filtrare cu filtru de medie

filtrare cu filtru Gaussian

Filtre normale (Gaussiene)

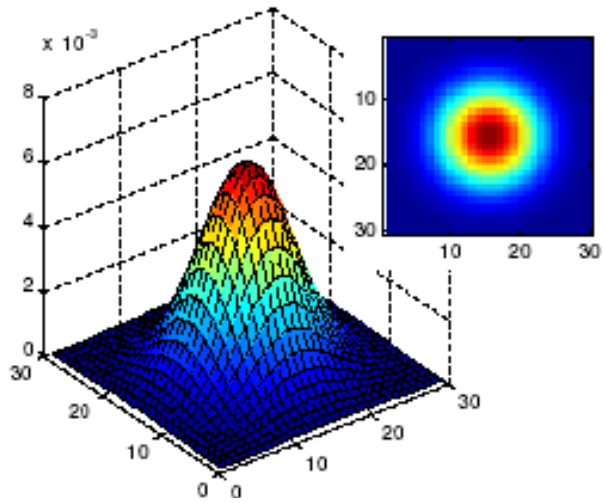
- Care sunt parametri ce definesc filtrul?

1. **Dimensiunea**
   - funcțiile normale (Gaussiene) au suport infinit (> 0), însă filtrele au dimensiune finită



σ = 5

10 x 10



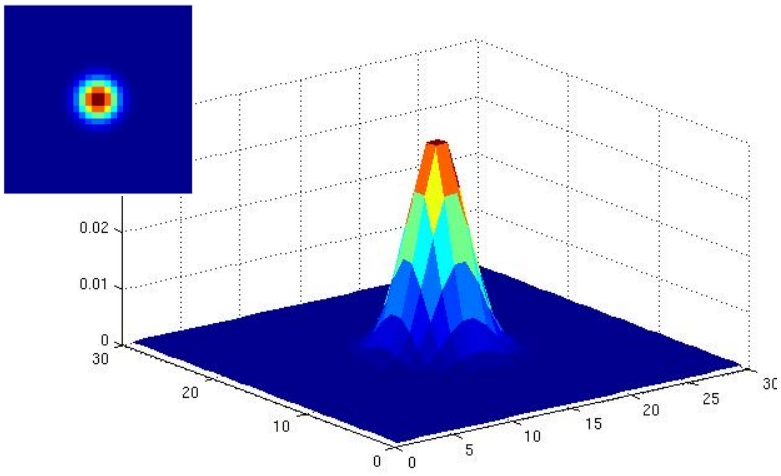
σ = 5

30 x 30

Filtre normale (Gaussiene)

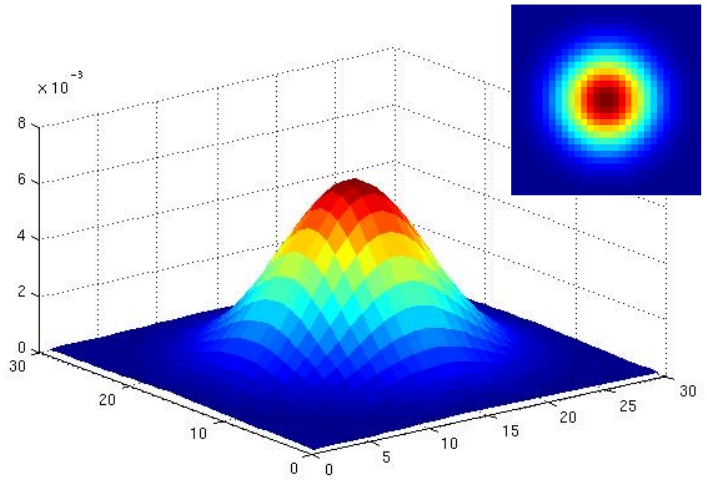
- Care sunt parametri ce definesc filtrul?

2. Deviația standard: cât de “neted” e filtrul



σ = 2

30 x 30



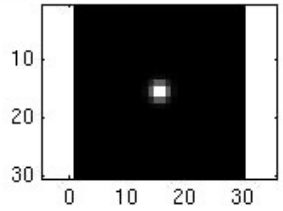
σ = 5

30 x 30

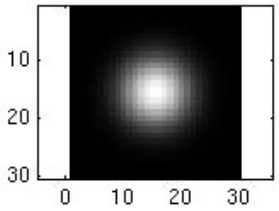
Slide adaptat după K. Grauman

Blurarea unei imagini cu un filtru Gaussian

Parametrul σ – controlează “netezimea” filtrului

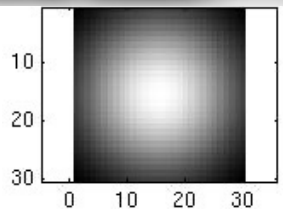


σ = 1



σ = 4

...



σ = 10

Slide adaptat după K. Grauman

Proprietăți ale filtrelor de blurare ("smoothing")

1. valori pozitive;

2. suma lor = 1 → pentru regiuni constante (toți pixelii au aceeași valoare → perete alb), output = input;

3. gradul de blurare/netezire este proporțional cu dimensiunea filtrului;

4. elimină regiunile de pixeli cu variantă mare în intensitate (frecvențe înalte); se mai numesc filtre "trece-jos" ("low-pass")

Proprietăți ale filtrelor de blurare ("smoothing")

ce se întâmplă (ce caracteristici are imaginea rezultată) dacă avem un filtru de blurare care nu este normalizat (suma elementelor filtrului ≠ 1)?

1	1	1
1	1	1
1	1	1

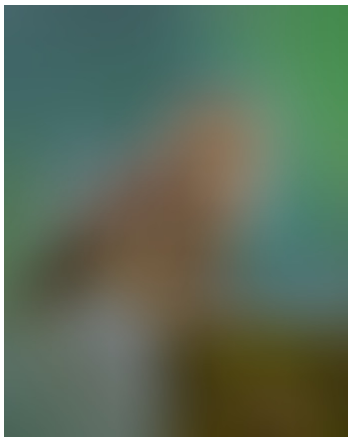
⊗

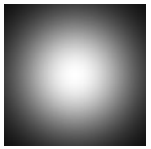
=

Dacă suma elementelor filtrului > 1 → imaginea se luminează
Dacă suma elementelor filtrului < 1 → imaginea se întunecă

Filtrarea imaginilor color

$$I(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix} \Rightarrow F \otimes I = \begin{bmatrix} F \otimes r \\ F \otimes g \\ F \otimes b \end{bmatrix}$$





σ = 30 pixels

Filtrarea imaginilor color

PYTHON: f = np.ones((1,9))/9

F = cv2.filter2D(img,-1,f)

f = [1/9 1/9 … 1/9]

Imagine inițială I

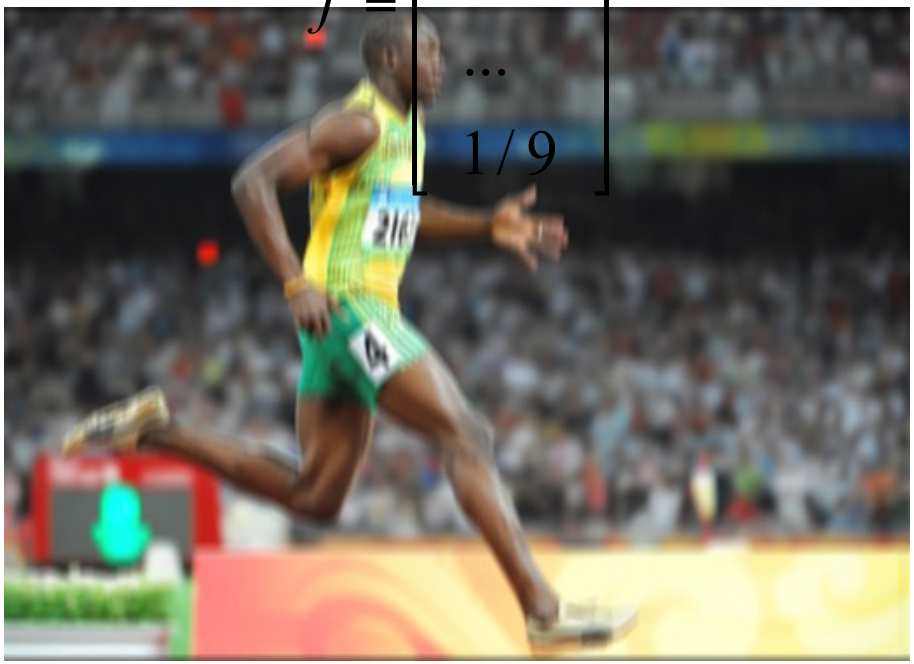
Imagine filtrată f * I

Filtrarea imaginilor color

PYTHON: f = np.ones((9,1))/9
F = cv2.filter2D(img,-1,f)

$$f = \begin{bmatrix} 1/9 \\ 1/9 \\ \dots \\ 1/9 \end{bmatrix}$$

Imagine inițială I



Imagine filtrată f * I

Filtrarea imaginilor color

PYTHON: f = np.ones((9,9))/81
F = cv2.filter2D(img,-1,f)

$$f = \begin{bmatrix} 1/81 & 1/81 & \dots & 1/81 \\ 1/81 & 1/81 & \dots & 1/81 \\ \dots & \dots & \dots & \dots \\ 1/81 & 1/81 & \dots & 1/81 \end{bmatrix}$$

Imagine inițială I

Imagine filtrată f * I

Filtrarea imaginilor color

PYTHON: f = np.ones((27,27))/(27*27)
F = cv2.filter2D(img,-1,f)

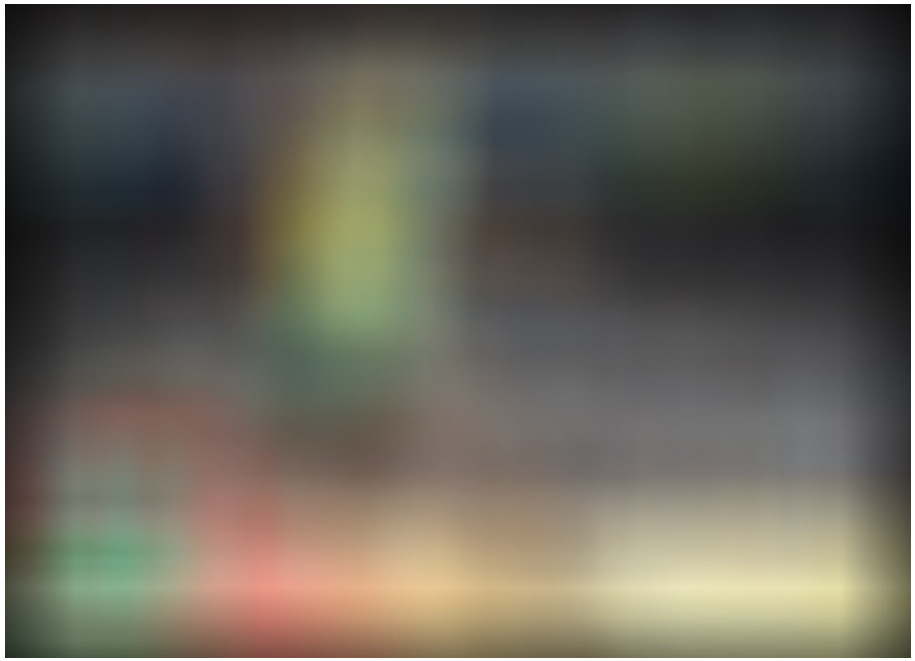
Imagine inițială I

Imagine filtrată f * I

Filtrarea imaginilor color

PYTHON: f = np.ones((81,81))/(81*81)
F = cv2.filter2D(img,-1,f)

Imagine inițială I



Imagine filtrată f * I

Filtrarea imaginilor color

PYTHON: f = np.ones((3,3))/4

F = cv2.filter2D(img,-1,f)

$$f = \begin{bmatrix} 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 \end{bmatrix}$$

Imagine inițială I

Imagine filtrată f * I

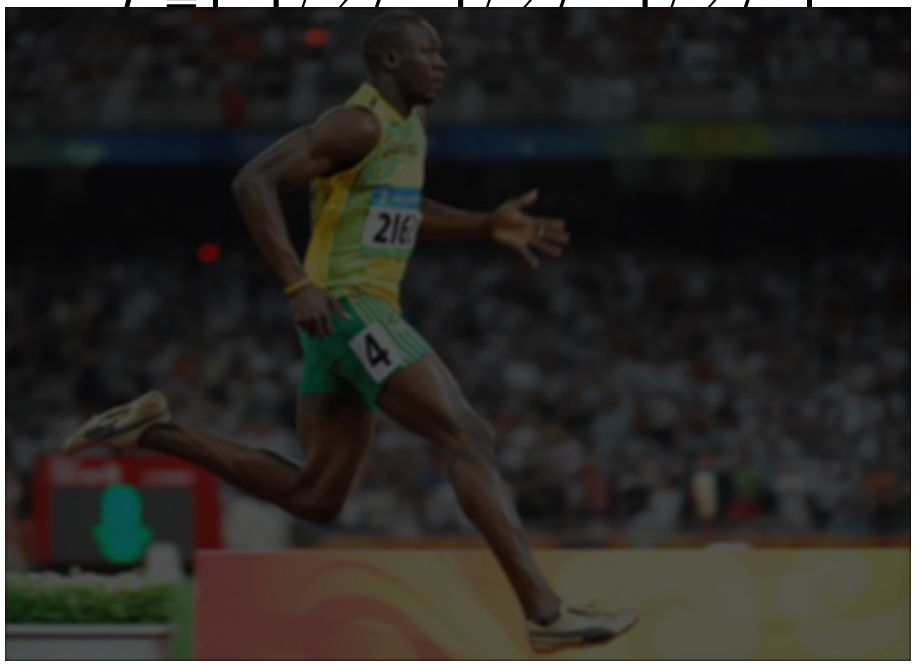
Filtrarea imaginilor color

PYTHON: f = np.ones((3,3))/27

F = cv2.filter2D(img,-1,f)

$$f = \begin{bmatrix} 1/27 & 1/27 & 1/27 \\ 1/27 & 1/27 & 1/27 \end{bmatrix}$$

Imagine inițială I



Imagine filtrată f * I

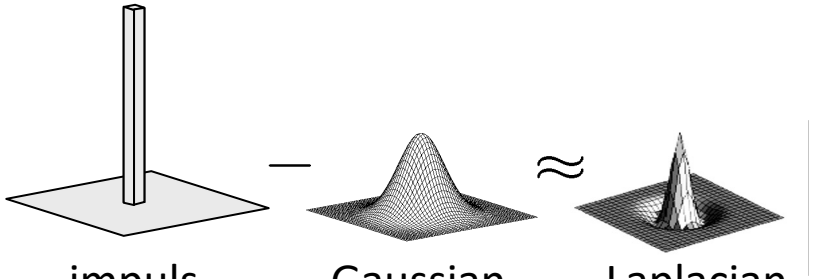
Aplicație: Imagini hibrid

filtru Gaussian
(low pass filter – obține frecvențe joase)

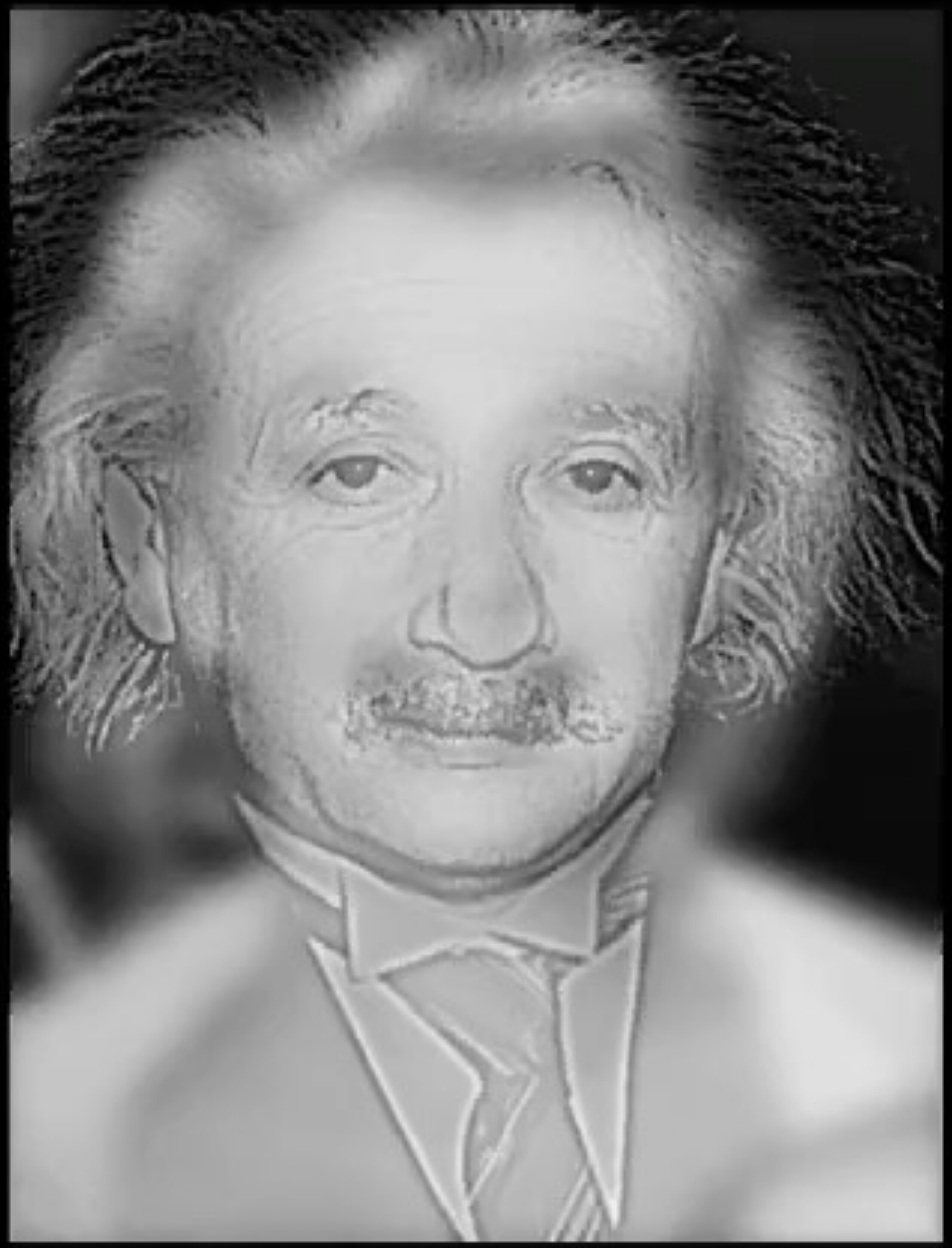
-

-

filtru Laplacian
(high pass filter – obține frecvențe înalte)



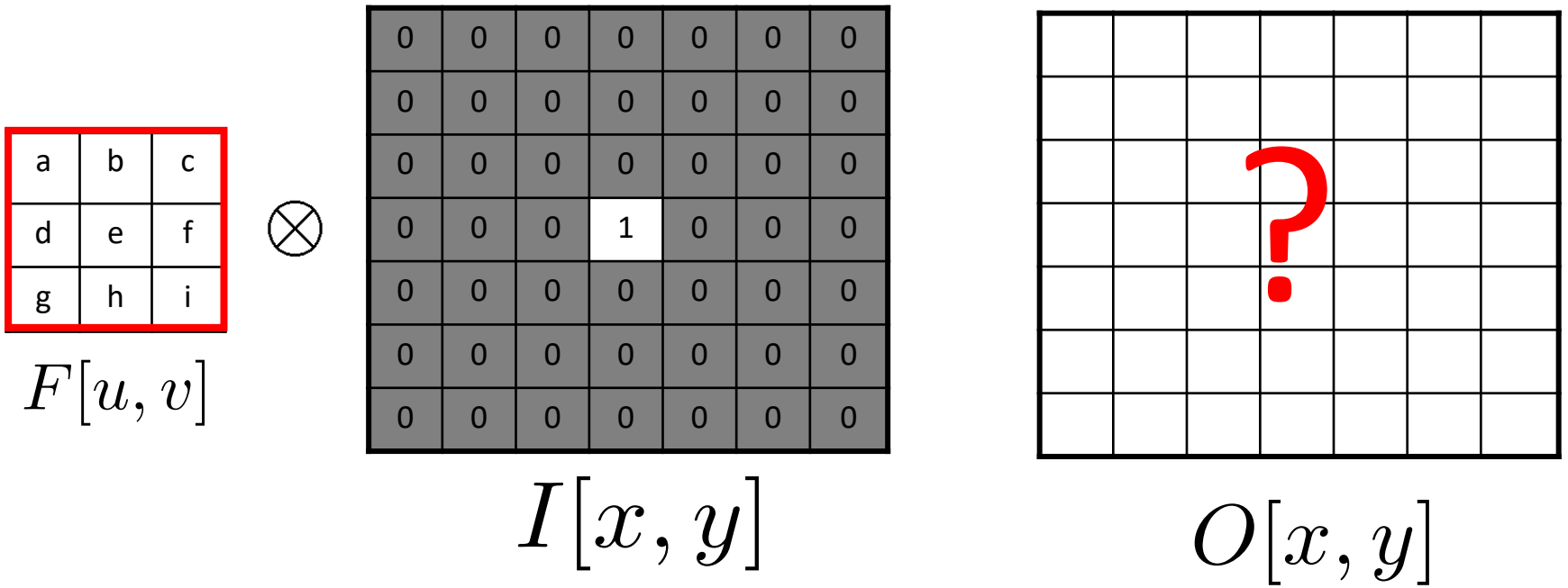
A. Oliva, A. Torralba, P.G. Schyns,
"Hybrid Images" SIGGRAPH 2006




BMX

Filtrarea unui semnal impuls

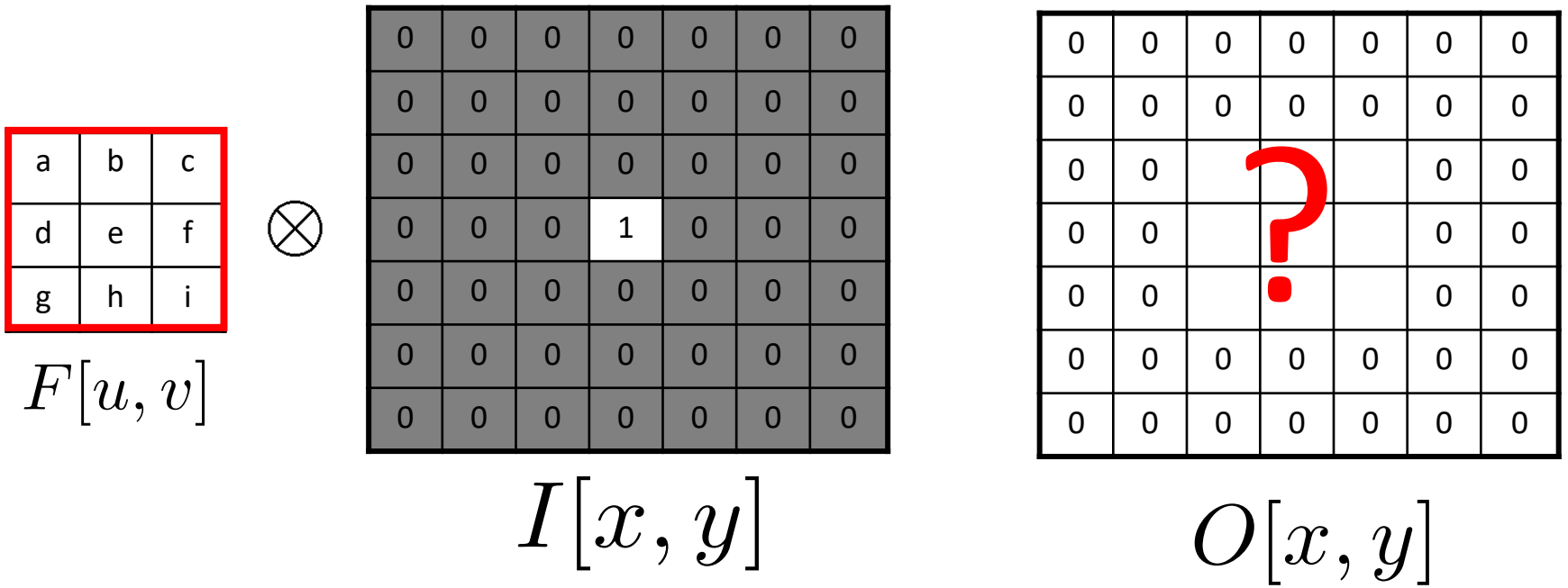
Ce rezultă în urma filtrării semnalului impuls (imaginea) I cu filtrul F ?



Slide adaptat după Steve Seitz

Filtrarea unui semnal impuls

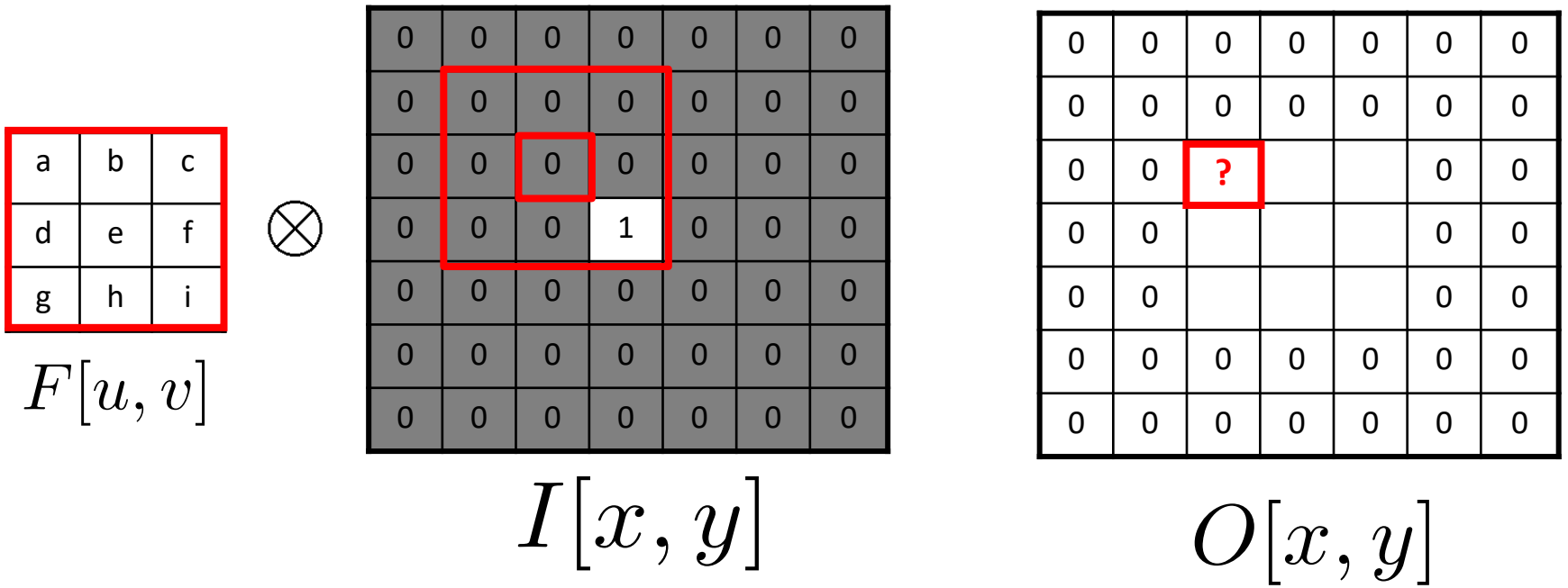
Ce rezultă în urma filtrării semnalului impuls (imaginea) *I* cu filtrul *F* ?



Slide adaptat după Steve Seitz

Filtrarea unui semnal impuls

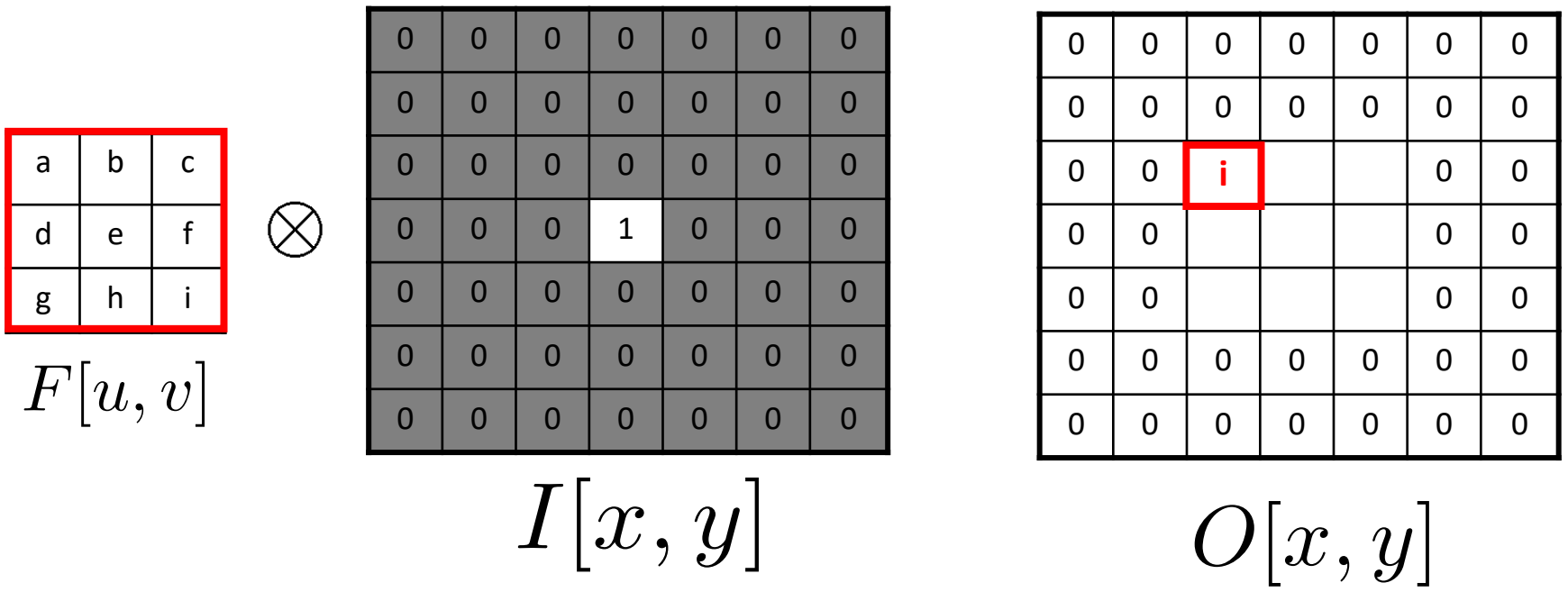
Ce rezultă în urma filtrării semnalului impuls (imaginea) *I* cu filtrul *F* ?



Slide adaptat după Steve Seitz

Filtrarea unui semnal impuls

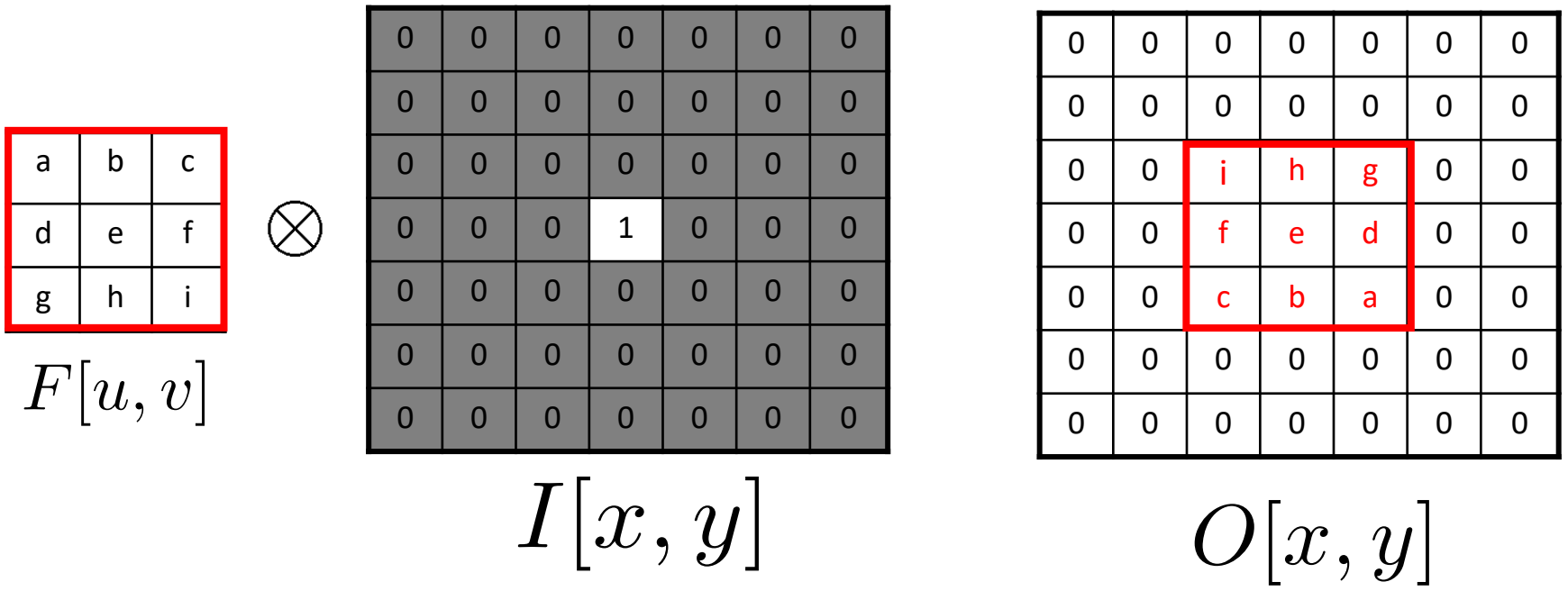
Ce rezultă în urma filtrării semnalului impuls (imaginea) *I* cu filtrul *F* ?



Slide adaptat după Steve Seitz

Filtrarea unui semnal impuls

Ce rezultă în urma filtrării semnalului impuls (imaginea) *I* cu filtrul *F* ?



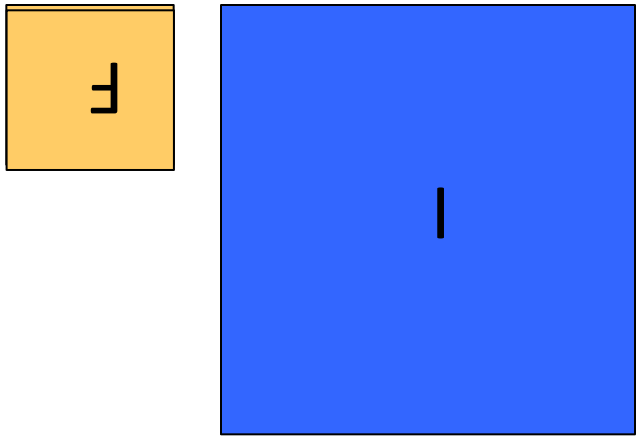
Convoluție

- Convoluție:
  - inversăm filtrul în ambele direcții (jos ⇔ sus, dreapta ⇔ stânga)
  - aplicăm corelația

$$O[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[u,v] I[i-u, j-v]$$

$$O = F \star I$$

notație pentru convoluție



Convoluție vs. corelație

Convoluție

$$O[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[u,v] I[i-u, j-v]$$

$$O = F \star I$$

Corelație

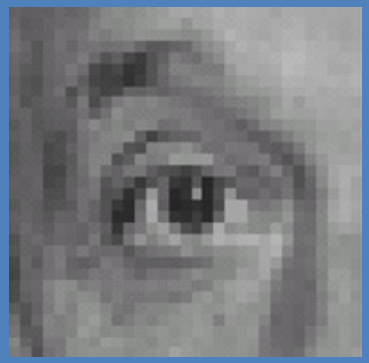
$$O[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[u,v] I[i+u, j+v]$$

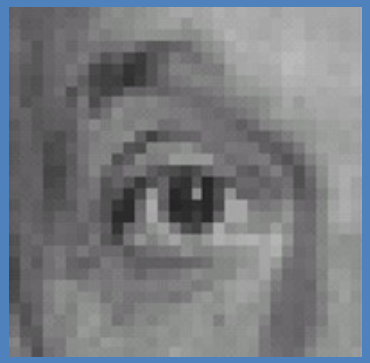
$$O = F \otimes I$$

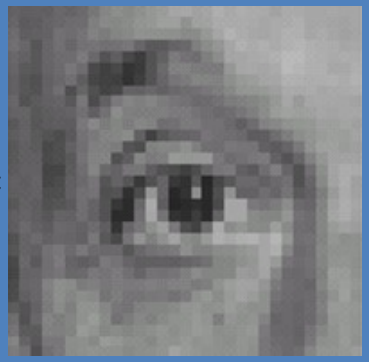
Pentru un filtru normal/de medie, cum diferă output-ul?

Pentru input = un semnal impuls, cum diferă output-ul?

Output = ?

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} * \text{  } = ?$$

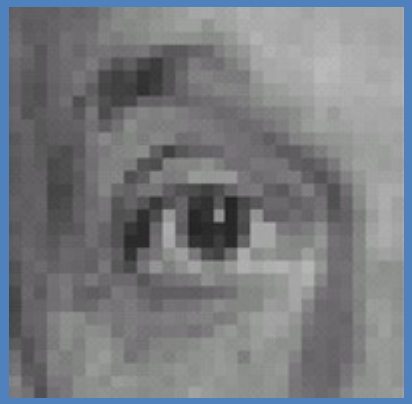
$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} * \text{  } = ?$$

$$\left[\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right] * \text{  } = ?$$

Filtre liniare

0	0	0
0	1	0
0	0	0

*



=

?

input

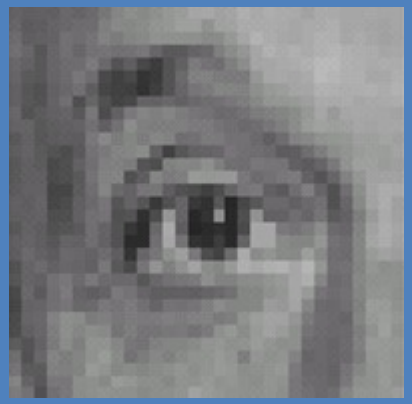
output

Slide adaptat după David Lowe

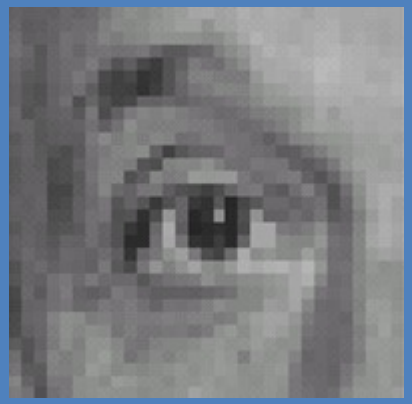
Filtre liniare

0	0	0
0	1	0
0	0	0

*



=

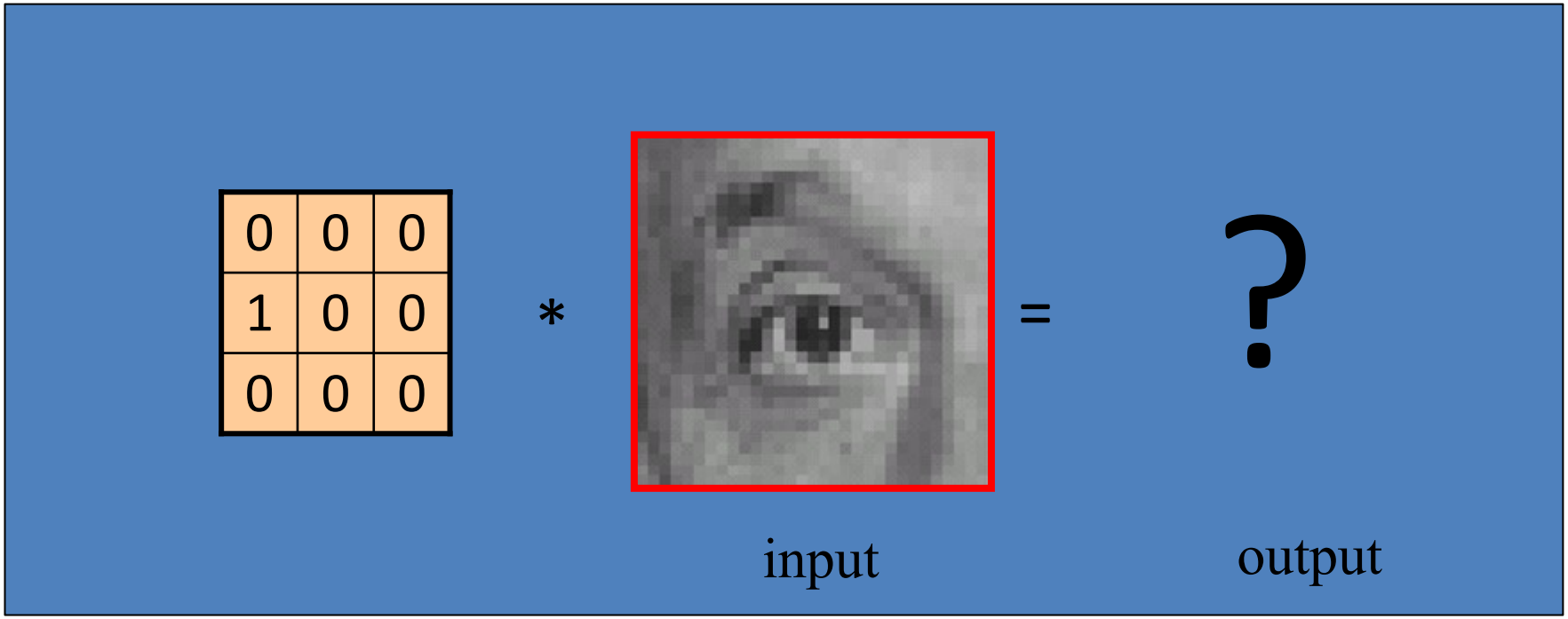


input

output (la fel)

Slide adaptat după David Lowe

Filtre liniare

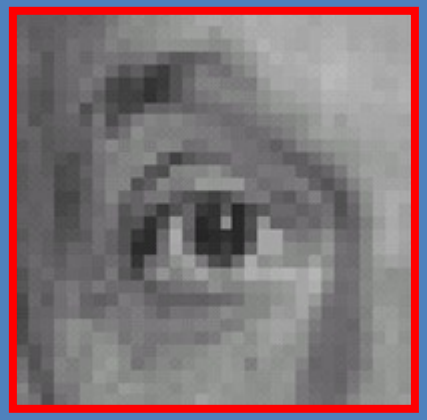


Slide adaptat după David Lowe

Filtre liniare

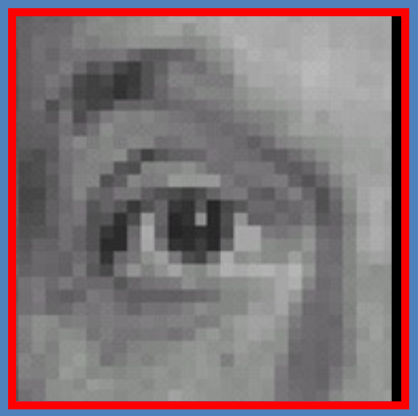
0	0	0
1	0	0
0	0	0

*



input

=



output: deplasat la stânga cu 1 pixel

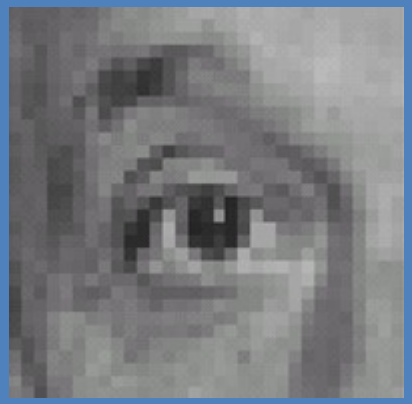
Slide adaptat după David Lowe

Filtre liniare

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

*



=

?

input

output

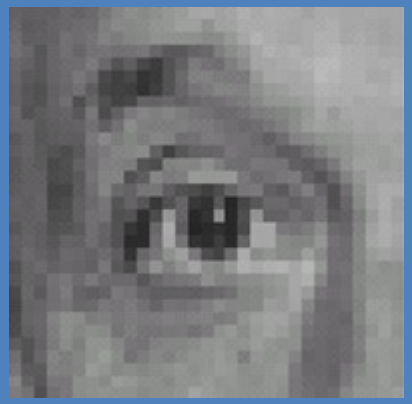
Slide adaptat după David Lowe

Filtre liniare

$\frac{1}{9}$

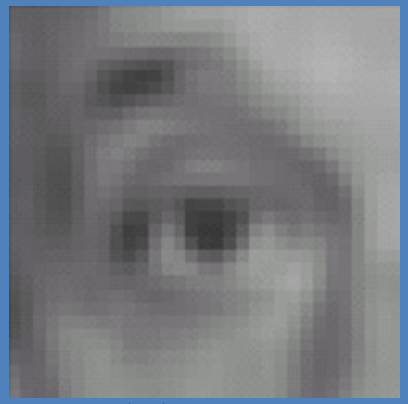
1	1	1
1	1	1
1	1	1

*



input

=

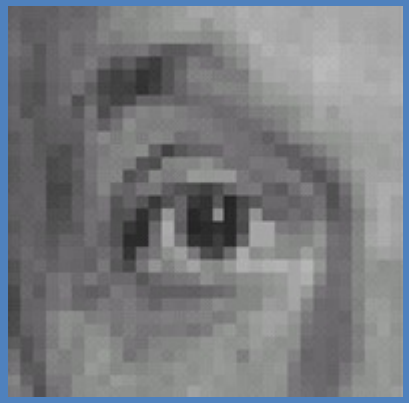


blurat
(cu filtru de medie)

Slide adaptat după David Lowe

Filtre liniare

$$\left[\begin{matrix}0&0&0\\0&2&0\\0&0&0\end{matrix} - \frac{1}{9}\begin{matrix}1&1&1\\1&1&1\\1&1&1\end{matrix}\right] * \text{input} = \text{output}$$



(Suma = 1)

?

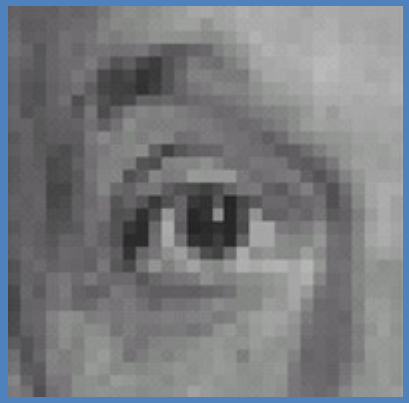
Slide adaptat după David Lowe

Filtre liniare

$\frac{1}{9}$

-1	-1	-1
-1	17	-1
-1	-1	-1

*



=

?

(Suma = 1)

input

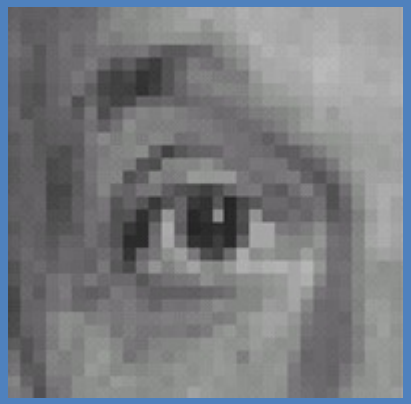
output

Slide adaptat după David Lowe

Filtre liniare

$\frac{1}{9}$

-1	-1	-1
-1	17	-1
-1	-1	-1

*  = 

(Suma = 1) input output

Filtru de accentuare - accentuează diferențele cu ajutorul mediilor locale

Slide adaptat după David Lowe

Reducerea de zgomot

imagine originală

“salt and pepper”

impuls

normal

Reducerea zgomotului "salt and pepper"

Aplicăm un filtru Gaussian de dimensiune $n \times n$

3×3

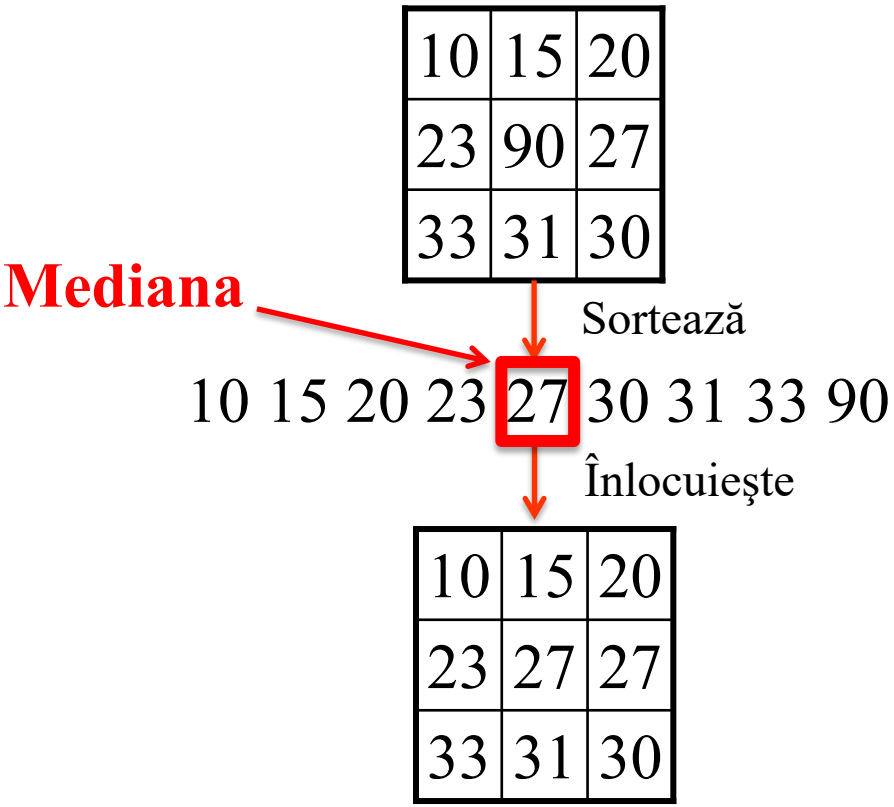
5×5

7×7

Zgomotul se elimină dar imaginea se deteriorează
(blurare + zgomotul se împrăștie la pixelii vecini)

Filtru median

- Un **filtru median** operează asupra unei ferestre prin selectarea intensității mediane



Slide adaptat după K. Grauman

Filtru median

- Un **filtru median** operează asupra unei ferestre prin selectarea intensității mediane

- Este liniar? (median(I_1+I_2) = median(I_1) + median(I_2))

mediana($\begin{matrix} 10 & 10 & 10 \\ 10 & 15 & 20 \\ 20 & 20 & 20 \end{matrix}$ + $\begin{matrix} 10 & 10 & 10 \\ 10 & 5 & 20 \\ 20 & 20 & 20 \end{matrix}$) = mediana($\begin{matrix} 20 & 20 & 20 \\ 20 & 20 & 40 \\ 40 & 40 & 40 \end{matrix}$) = 20

mediana($\begin{matrix} 10 & 10 & 10 \\ 10 & 15 & 20 \\ 20 & 20 & 20 \end{matrix}$) + mediana($\begin{matrix} 10 & 10 & 10 \\ 10 & 5 & 20 \\ 20 & 20 & 20 \end{matrix}$) = 15 + 10 = 25

- NU este liniar

Filtrul median vs. filtrul Gaussian

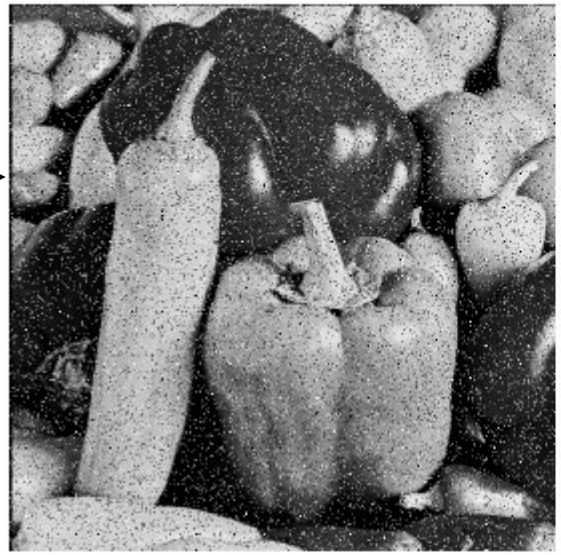
	3×3	5×5	7×7
filtru Gaussian			
zgomot "salt and peper"			
filtru median			

Zgomotul se elimină imaginea cu un filtru median 3 x 3. Pentru vecinătăți mai mari imaginea se deteriorează (se uniformizează).

Filtrul median

zgomot

“salt and peper”



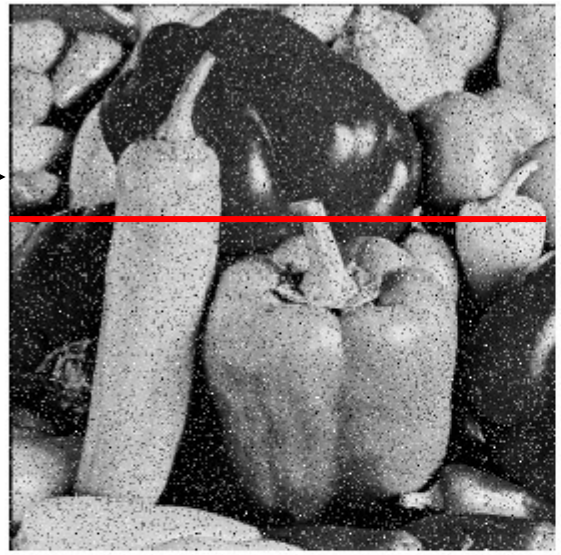
după aplicarea filtrului median

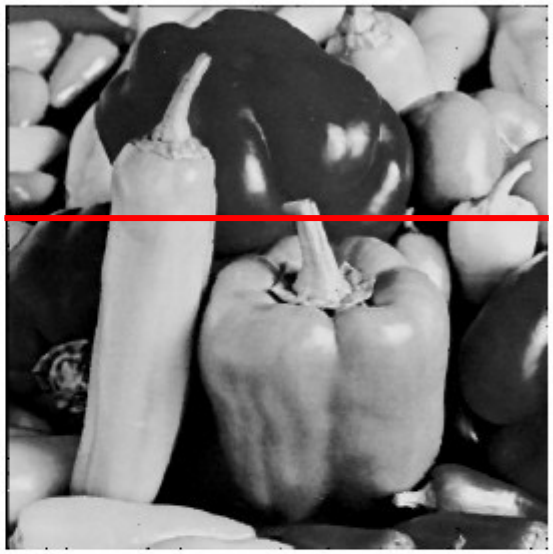
Slide adaptat după M. Herbert

Filtrul median

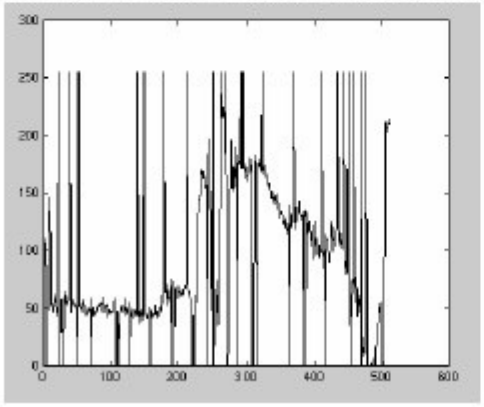
zgomot

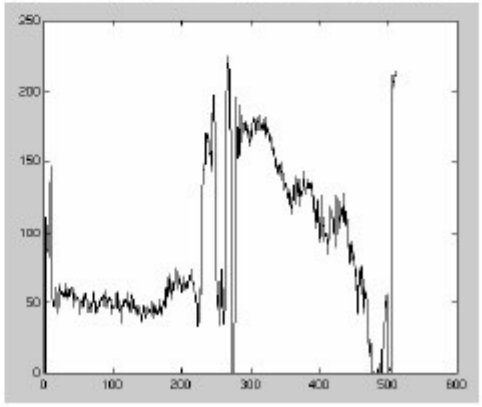
“salt and peper”





după aplicarea filtrului median





Plotăm intensitatea pentru pixelii (de pe linia roșie) din imagini

Slide adaptat după M. Herbert