Concepte și aplicații în Vederea Artificială

Bogdan Alexe

bogdan.alexe@fmi.unibuc.ro

Cursul 2

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

Cursul trecut

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

Cursul de azi

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

normal

- Filtrarea neliniară
  - filtrul median

Formarea imaginilor

Formarea imaginilor

Slide adaptat după Derek Hoiem

Camere digitale

http://electronics.howstuffworks.com/digital-camera.htm

Senzori de imagine:

1. CMOS (complementary metal oxide conductor)
2. CCD (charge couple device)

CMOS

CCD

Slide adaptat după Steve Seitz

Imagini digitale

Imagine proiectată pe senzorul de imagine

Imagine rezultată în urma discretizării
- eşantionare - discretizează spațiul în pixeli
- cuantizare - discretizează luminozitatea

Imagine digitală = matrice bidimensională; elementele ei se numesc pixeli (pixel = picture element)

Slide adaptat după Derek Hoiem

Imagini digitale color

Senzor de imagine Filtru Bayer

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

RGB2GRAY

Color

Grayscale (tonuri de gri)

gray = 0.2989 * R + 0.5870 * G + 0.1140 * B

coeficienți determinați pe baza percepției vizuale umane

https://en.wikipedia.org/wiki/Color_vision

Imagini privite ca funcții

- Imaginea $I : \mathbf{R^2} \rightarrow \mathbf{R}$, $I(x, y)$ - intensitatea la (x,y)
- $I : [\mathrm{a\ b}] \times [\mathrm{c\ d}] \rightarrow [i_{min}\ \ i_{max}]$ (interval + intensitate mărginite)
- Imagini color: $I(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}$

Slide adaptat după Steve Seitz

Imagini digitale

- I - matrice 2D cu valori întregi
- Intensități {0, 1, …, 255}
  - luminozitate 0 = negru
  - luminozitate 255 = alb

y →

x ↓

62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

Slide adaptat după Steve Seitz

Imagini în Matlab

- Imaginile sunt reprezentate ca matrice
- “img” – imagine RGB de dimensiuni N x M
  - img(1,1,1) = valoarea pixelului din stânga-sus în canalul R
  - img(y, x, c) = y pixeli în jos, x pixeli în dreapta, canalul c
  - img(N, M, 3) = valoarea pixelului din dreapta-jos în canalul B
- imread(filename) returnează o imagine uint8 (valori 0 – 255)

Slide adaptat după Derek Hoiem

Proiectul 1

- termen limită pentru prima temă: duminică, 28 octombrie!
- e nevoie de cunoștințe de procesarea imaginilor în Matlab + programare ☺

Imagini mozaic

imagine de referință

+

colectie de imagini (piese)
de dimensiuni reduse
28 x 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**

1	2		100

3. **număr piese în lățime ale mozaicului (implicit = 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 pozitie:**
   - **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 referinta

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 referinta

imagine mozaic

Criterii

- culoarea medie cea mai apropiată

imagine de referinta

imagine mozaic

Vecini – piese diferite

Vecini – piese diferite

Piese hexagonale

Piese hexagonale

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)

Distributia normală 1D (clopotul lui Gauss)

MATLAB:

X = randn(100,200); %generează o matrice 100 x 200 de variabile cu
% distributie normala standard:
%medie = 0, deviatie standard = 1

X = mu + sigma * randn(100,200); %medie = mu, deviatie standard = sigma

Regula 68-95-99.7

Zgomot normal – efectul lui σ

sigma = 1

sigma = 5

sigma = 25

MATLAB:

```
for sigma = [1 5 25]
        zgomot = sigma*randn(size(img));%medie = 0, dev. std. = sigma
        newImg = uint8(round(double(img) + zgomot));
        figure, imshow(newImg);
end
```

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

$I[x, y]$

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								

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							

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						

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					

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 x 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 x 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: $O = F \otimes I$

(-k,-k)				(-k,+k)
		(0,0)		
(+k,-k)				(+k,+k)

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.

Filtru **MATLAB** **Vecinătate pixel**

sum

(-k,-k)				(-k,+k)
		(0,0)		
(+k,-k)				(+k,+k)

.*

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

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)?

$F[u,v]$ $\otimes$ $I[x,y]$ $O[x,y]$

$\frac{1}{9}$

1	1	1
1	?	1
1	1	1

filtru de medie
(filtru pătrat)

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

	0	10	20	30	30				

$$O = F \otimes I$$

Slide adaptat după Steve Seitz

Blurarea (netezirea) unei imagini cu un filtru de medie

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

Imagine inițială

Imagine filtrată

Dimensiune output

- Care este dimensiunea imaginii filtrate?

Dimensiune output

- Care este dimensiunea imaginii filtrate?
- MATLAB: opțiuni pentru filtrare
  - ‘full’: dim(O) = dim(I) + dim(F)
  - ‘same’: dim(O) = dim(I)
  - ‘valid’: dim(O) = dim(I) – dim(F)

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 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

Pixelii din afara imaginii

- Ce valori au pixelii din afara imaginii ?
  - extrapolăm valori
  - opțiuni în MATLAB:

1. imfilter(I, f, 0)
2. imfilter(I, f, ‘circular’)
3. imfilter(I, f, ‘replicate’)
4. imfilter(I, f, ‘symmetric’)

Slide adaptat după S. Marschner

Filtru normal (Gaussian)

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

Acest filtru aproximează distribuția normală în 2D:

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

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$F[u,v]$

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:

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

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$F[u,v]$

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

$I[x,y]$

Slide adaptat după Steve Seitz

Blurarea unei imagini

Filtru de medie

$$\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

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

muchie verticală

Filtru normal

$$\frac{1}{16}\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix}$$

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

OBSERVAȚII:

1. **se elimină regiunile din imagine cu schimbări bruște de intensitate (frecvențe înalte) - muchiile**

2. **se micșorează distanțele dintre pixeli cu intensitate mare și pixeli cu intensitate mică**

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

Slide adaptat după Steve Seitz

Blurarea unei imagini

filtru Gaussian:
alb = valoare mare,
negru = valoare mică

filtru de medie

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

Slide adaptat după K. Grauman

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

Matlab

```
>> fsize = 30;
>> sigma = 5;
>> f = fspecial('gaussian', fsize, sigma);
```

```
>> mesh(f);
```

```
>> imagesc(f);
```

```
>> outim = imfilter(im, f); % corelatie
>> imshow(outim);
```

im

→

outim

Slide adaptat după K. Grauman

Blurarea unei imagini cu un filtru Gaussian

Parametrul σ – controlează “netezimea” filtrului

σ = 1

σ = 4

...

σ = 10

```
fsize = 30;
for sigma = 1:3:10
   f = fspecial('gaussian‘, fsize, sigma);
   out = imfilter(im, f);
   imshow(out);
   pause;
end
```

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

$\otimes$

=

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

Aplicație: Imagini hibrid

filtru Gaussian
(low pass filter – obține frecvențe joase)

-

-

filtru Laplacian
(high pass filter – obține frecvențe înalte)

impuls Gaussian Laplacian

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

BMX