Lucrarea 4

Procesarea imaginilor digitale

Aplicații ale convoluției bidimensionale în procesarea de imagini. Proiectarea, aplicarea și utilitatea diferitelor tipuri de filtre de imagine.

1. Noțiuni teoretice

1.1 Reprezentarea imaginilor

Imaginile digitale sunt structuri de date de tip matrice. Ele au o anumită lățime (coloane) și înălțime (rânduri), iar elementele ce compun imaginea poartă denumirea de pixeli și conțin informații despre intensitatea culorii. În funcție de această informație se disting următoarele tipuri de imagini:

- Imagini în scală de gri;
- Imagini color;
- Imagini binare.

În cazul imaginilor în scală de gri, intensitatea pixelilor descrie nuanțe de gri cuprinse între alb și negru. Pentru imaginile color, fiecare pixel este definit de trei intensități separate, câte una pentru fiecare culoare primară de lumină: roșu, verde și albastru.

În ambele tipuri de imagine descrise anterior, valorile culorilor sunt reprezentate pe 8 biți. Acest mod de reprezentare oferă, deci, 256 de niveluri pentru intensitatea oricărei culori sau nivel de gri. Prin extensie rezultă că este posibil să reprezentăm aproximativ 16,7 milioane de culori diferite folosind un singur byte pentru fiecare culoare primară.

Imaginile binare sunt imagini ce conțin doar alb sau negru, pixelii putând lua doar două valori: 0 sau 1.

Cel mai utilizat pachet pentru Python folosit în aplicațiile de procesare de imagine este *OpenCV*. În Thonny, acesta se poate adăuga căutând *opencv-python* în fereastra de gestionare a pachetelor.

```
import cv2
import matplotlib.pyplot as plt

I=cv2.imread('Imagine.jpg',1) #incarca Imagine.jpg in variabila I

I= cv2.cvtColor(I, cv2.COLOR_BGR2RGB) #modifica reprezentarea imaginii din BGR
in RGB

plt.imshow(I) #afisare imagine
plt.title('Imagine')
plt.axis('off') #sterge axele
plt.show()
```

Figura 1. Exemplu de imagine afișată în Python

Intensitățile culorilor primare sunt salvate separat în straturi numite canale (channels). Reprezentarea culorilor primare pe cele trei canale se face sub formă de imagine în scală de gri, unde valorile mici („negru”) indică o intensitate redusă a culorii, iar valorile mari („alb”) o intensitate ridicată. Exemplul de mai jos prezintă această descompunere pe canale:

```
import cv2
import matplotlib.pyplot as plt

I=cv2.imread('Imagine.jpg',1)

I=cv2.cvtColor(I, cv2.COLOR_BGR2RGB)

R=I[:,:,0] #Salveaza componenta 0 pentru fiecare element din matrice
G=I[:,:,1] #Salveaza componenta 1 pentru fiecare element din matrice
B=I[:,:,2] #Salveaza componenta 2 pentru fiecare element din matrice

plt.figure(0)
plt.subplot(2,2,1)
plt.imshow(R, cmap='gray', vmin=0, vmax=255)
plt.title('Canal R')
plt.axis('off')

plt.subplot(2,2,2)
plt.imshow(G, cmap='gray', vmin=0, vmax=255)
plt.title('Canal G')
plt.axis('off')
```

```
plt.subplot(2,2,3)
plt.imshow(B, cmap='gray', vmin=0, vmax=255)
plt.title('Canal B')
plt.axis('off')

plt.subplot(2,2,4)
plt.imshow(I)
plt.title('Imagine color')
plt.axis('off')
plt.tight_layout()
plt.show()
```

Figura 2. Canalele RGB pentru o imagine color

1.2 Histograme de imagini

O reprezentarea de tip histogramă indică distribuția datelor dintr-un set în funcție de un anumit parametru. În cazul imaginilor, histograma prezintă distribuția pixelilor în funcție de intensitate. În mod normal se reprezintă histograma generală a unei imagini, cea care nu ține cont de intensitatea individuală a fiecărei culori primare. Se pot însă realiza histograme și pentru fiecare canal individual (Figura 3).

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

I=cv2.imread('Imagine.jpg',1)
I= cv2.cvtColor(I, cv2.COLOR_BGR2RGB)
R=I[:,:,0]
```

```
G=I[:,:,1]
B=I[:,:,2]

plt.subplot(2,2,1)
plt.hist(R.ravel(),256,[0,256])
plt.title('Histograma Rosu')

plt.subplot(2,2,2)
plt.hist(G.ravel(),256,[0,256])
plt.title('Histograma Verde')

plt.subplot(2,2,3)
plt.hist(B.ravel(),256,[0,256])
plt.title('Histograma Albastru')

plt.subplot(2,2,4)
plt.hist(I.ravel(),256,[0,256])
plt.title('Histograma totala')
plt.tight_layout()
plt.show()
```

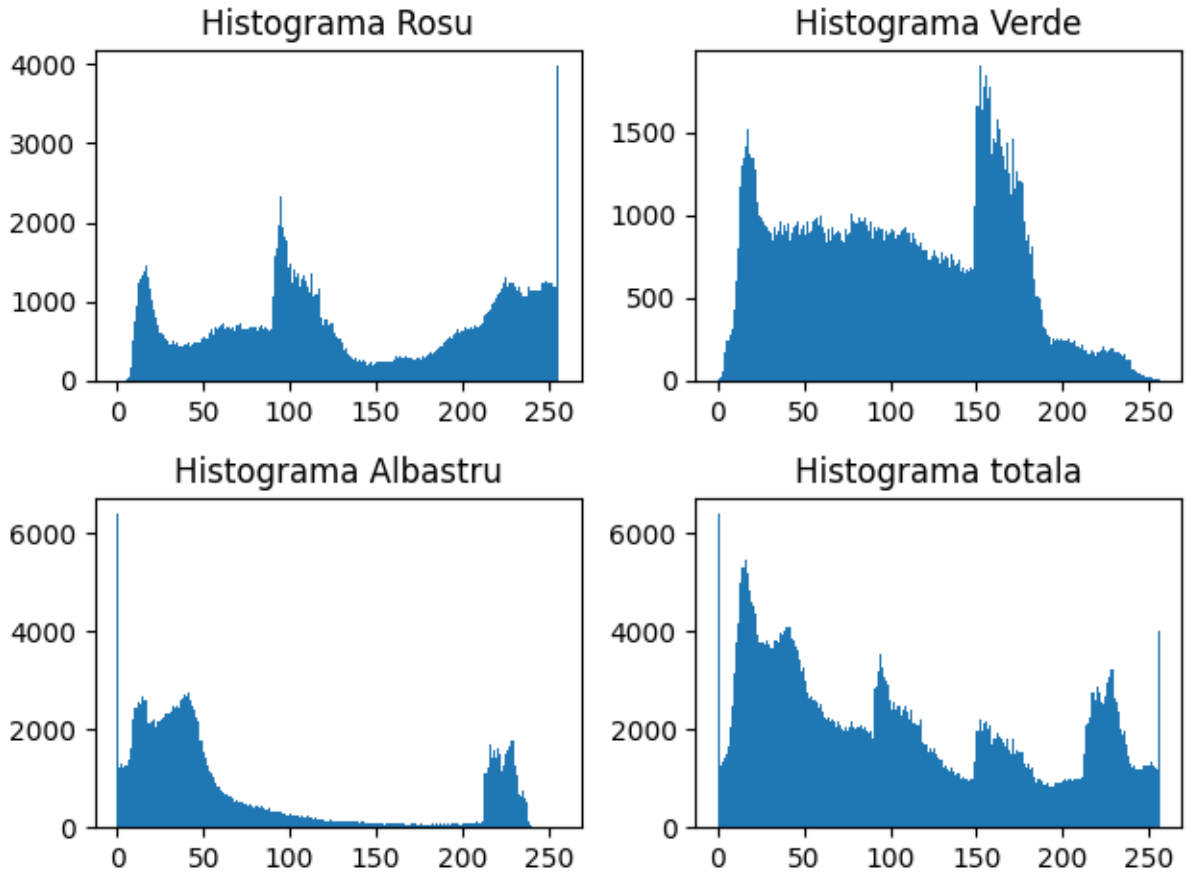

Figura 3. Exemplu de histograme pentru canalele RGB și pentru imaginea totală

1.3 Convoluția matricilor

Asemenea multor aplicații din domeniul procesării digitale a semnalelor, prelucrarea imaginilor digitale se bazează pe operația de convoluție. În spațiul bidimensional, operația de convoluție ia următoarea formă:

$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j] \quad (1)$$

Practic, această convoluție generează o matrice pe baza altor două matrici. Matricea h poartă denumirea de nucleu (kernel), iar matricea x este matricea de intrare. Nucleul „glisează” peste fiecare element al matricei x, și generează, pe rând, fiecare element al matricei de ieșire y ().

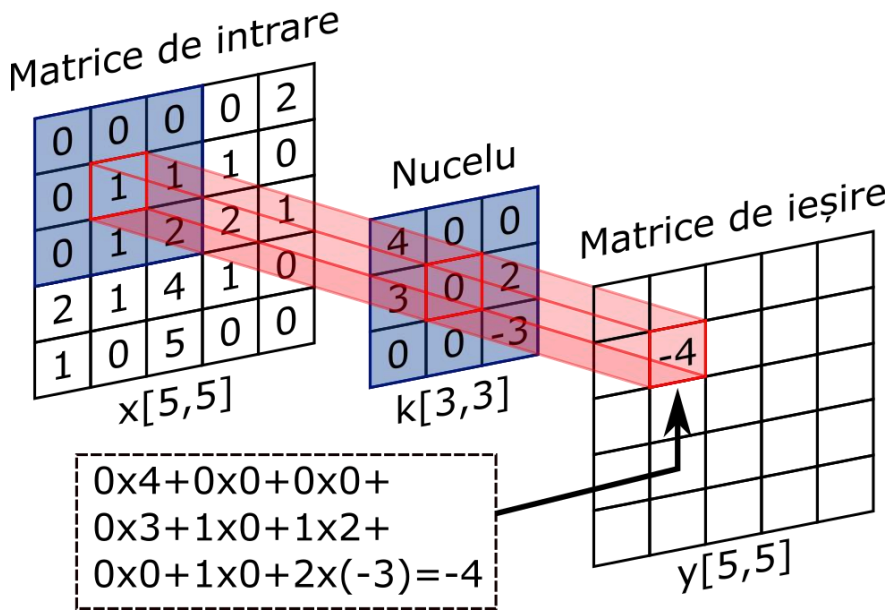

Figura 4. Exemplu de convoluție a matricilor

1.4 Filtre de imagine

Filtrele de imagine sunt aplicații ale convoluției descrise anterior. Ceea ce definește filtrul este nucleul. Se pot genera o infinitate de filtre prin schimbarea dimensiunilor și valorilor elementelor din nucleu.

Exemplificarea filtrelor de imagine este mai ușoară de realizat pe imagini în scală de gri. Transformarea unei imagini color într-una doar cu nuanțe de gri se face prin funcția *cv2.cvtColor(nume_imagine, cv2.COLOR_BGR2GRAY)* sau *cv2.cvtColor(nume_imagine, cv2.COLOR_RGB2GRAY)*, în funcție de ordinea canalelor imaginii.

Cele mai simple filtre sunt cele de estompare (blur). Există două tipuri principale de estompare: normală și gaussiană. Ele diferă prin ponderile pe care le au pixelii adiacenți în

calculele pentru noua valoare a pixelului central. Estomparea normală folosește aceiași pondere pentru toți pixelii adiacenți, în timp ce estomparea gaussiană oferă o pondere mai mare pixelilor imediat învecinați de cel central și mai mică celor mai îndepărtați (Tabelul 1).

Tabelul 1. Nuclee pentru diferite tipuri de filtre de estompare

Estompare normală	Estompare gaussiană 3x3	Estompare gaussiană 5x5
$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\1 & 1 & 1\\1 & 1 & 1\end{bmatrix}$	$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\2 & 4 & 2\\1 & 2 & 1\end{bmatrix}$	$\frac{1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\4 & 16 & 24 & 16 & 4\\6 & 24 & 36 & 24 & 6\\4 & 16 & 24 & 16 & 4\\1 & 4 & 6 & 4 & 1\end{bmatrix}$

Exemplul următor ilustrează modul în care sunt aplicate filtrele definite în tabelul anterior.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

I=cv2.imread('Imagine.jpg',1) #incarca Imagine.jpg in variabila I
BW= cv2.cvtColor(I, cv2.COLOR_BGR2GRAY) #schimba imaginea in scala de gri

N = 1/9*np.ones([3,3]) #nucleu normal
G3 = 1/16*np.array([[1, 2, 1],[2, 4, 2],[1, 2, 1]])#nucleu gaussian 3x3
G5 = 1/256*np.array([[1, 4, 6, 4, 1],[4, 16, 24, 16, 4],[6, 24, 36, 24, 6],[4,
16, 24, 16, 4],[1, 4, 6, 4, 1]])#nucleu gaussian 5x5

#convolutie
BWN = cv2.filter2D(src=BW, ddepth=-1, kernel=N)
BWG3 = cv2.filter2D(src=BW, ddepth=-1, kernel=G3)
BWG5= cv2.filter2D(src=BW, ddepth=-1, kernel=G5)

#afisare
plt.figure(2)
plt.subplot(2,2,1)
plt.imshow(BW,cmap='gray', vmin=0, vmax=255)
plt.title('Original')
plt.axis('off')
plt.subplot(2,2,2)
plt.imshow(BWN,cmap='gray', vmin=0, vmax=255)
plt.title('Estompare normala')
plt.axis('off')
plt.subplot(2,2,3)
plt.imshow(BWG3,cmap='gray', vmin=0, vmax=255)
plt.title('Estompare gaussiana 3x3')
plt.axis('off')
plt.subplot(2,2,4)
plt.imshow(BWG5,cmap='gray', vmin=0, vmax=255)
plt.title('Estompare gaussiana 5x5')
plt.axis('off')
plt.tight_layout()
plt.show()
```

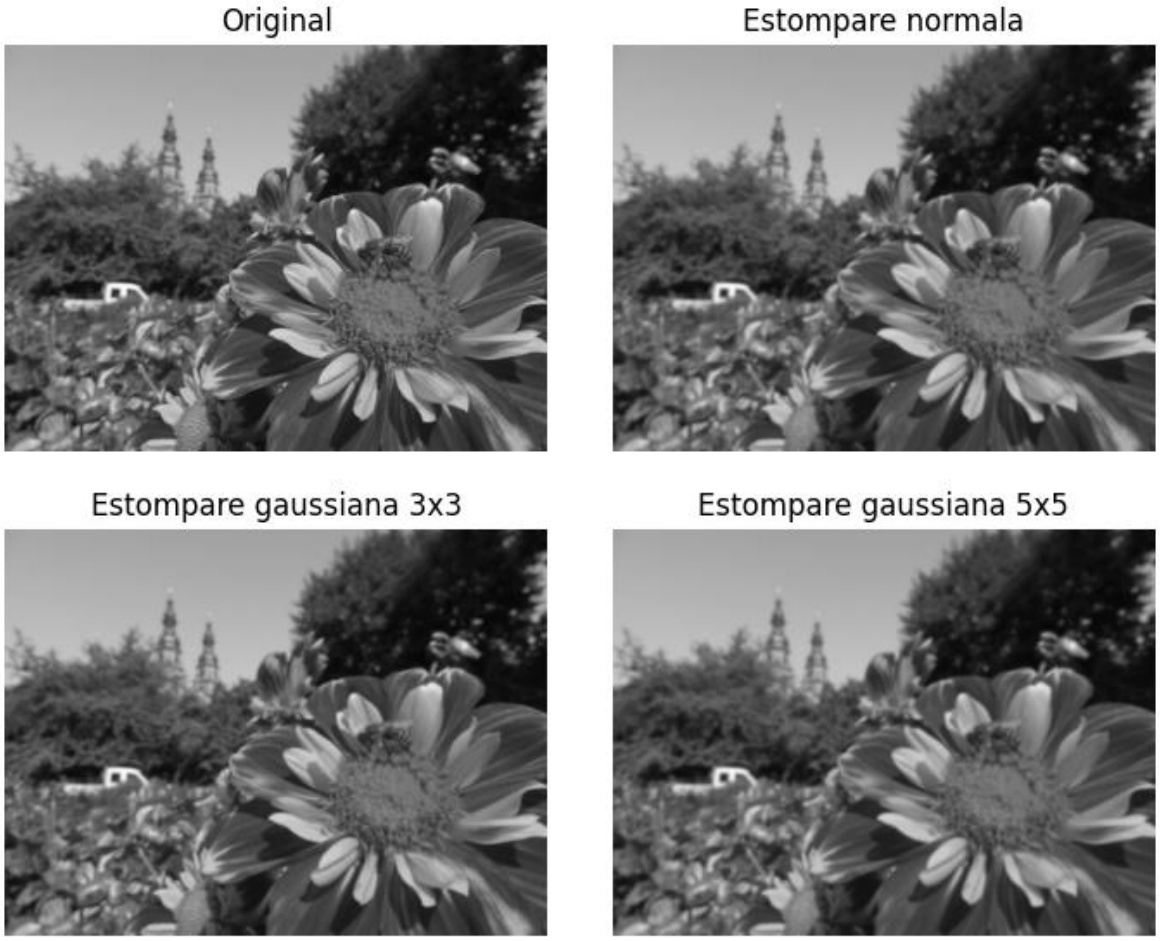

Figura 5. Exemplu de aplicare a filtrelor de estompare

Tabelul următor conține nucleele unor filtre des utilizate în procesarea de imagini. Acestea sunt disponibile în majoritatea programelor software de editare a imaginilor.

Tabelul 2. Exemple de alte filtre de imagine

Denumire filtru	Nucleu	Efect
Unitate (Identity)	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	Lasă imaginea neschimbată
Ascuțire (Sharpen)	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	Opusul filtrului de estompare
Contur (Outline)	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	Păstrează doar conturul elementelor din imagine
Reliefare (Emboss)	$\begin{bmatrix} -2 & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix}$	Accentuează diferențele dintre pixeli, oferind iluzia adâncimii în imagine

Sobel – dreapta	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$	Păstrează doar marginile orientate spre dreapta
Sobel – stânga	$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	Păstrează doar marginile orientate spre stânga
Sobel – sus	$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$	Păstrează doar marginile orientate în sus
Sobel – jos	$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$	Păstrează doar marginile orientate în jos

Folosirea filtrelor pe imagini color necesită aplicarea acestora pe fiecare canal de culoare în parte. Mai exact, un filtru color este compus din trei nuclee care pot fi identice sau diferite, în funcție de efectul dorit

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

I=cv2.imread('Imagine.jpg',1)
I= cv2.cvtColor(I, cv2.COLOR_BGR2RGB)

FR = np.array([[0, 0.1, 0],[0.1, 0.1, 0.1],[0, 0.1, 0]])
FG = np.array([[0.5, 0, 0.5],[0, -1, 0],[0.5, 0, 0.5]])
FB = 1/16*np.array([[1, 2, 1],[2, 4, 2],[1, 2, 1]])

F=np.zeros((I.shape)) #defineste un array cu elemente 0 de dimensiunile
variabilei I

F[:,:,0] = cv2.filter2D(src=I[:,:,0], ddepth=-1, kernel=FR)
F[:,:,1] = cv2.filter2D(src=I[:,:,1], ddepth=-1, kernel=FB)
F[:,:,2] = cv2.filter2D(src=I[:,:,2], ddepth=-1, kernel=FG)

plt.imshow(F.astype('uint8')) #pentru afisarea corecta trebuie ca elementele
imaginii sa fie de tip uint8
plt.axis('off')
plt.show()
```

Figura 6. Imaginea color filtrată

2. Modul de lucru

Se parcurg următoarele cerințe, notându-se comanda executată și rezultatul acesteia.

1) Încărcați o imagine color și reprezentați cele trei canale de culoare primară.
2) Realizați reprezentările de tip histogramă pentru cele trei canale de culoare primară și pentru întreaga imagine.
3) Transformați imaginea în scală de gri și realizați afișarea acesteia.
4) Aplicați un filtru gaussian 3x3 imaginii.
5) Aplicați un filtru de ascuțire (sharpen) imaginii.
6) Aplicați un filtru de contur imaginii.
7) Aplicați un filtru definit de voi imaginii.

3. Lucrarea 4 în MATLAB/Octave

Această secțiune cuprinde exemplele de cod prezentate anterior, dar adaptate pentru a fi utilizate în MATLAB/Octave

4.1 Încărcarea imaginilor

Imaginile pot fi încărcate ca variabile în Octave/MATLAB folosind comanda *imread("imagine.extensie")* și afișate prin comanda *imshow(imagine)*. **Atenție!** Înainte de a încărca orice imagine sau fișier extern trebuie să ne asigurăm că ne aflăm în directorul corect. Navigarea se face prin intermediul butonului *browse directories* din bara de navigare de sus.

```
I=imread("Imagine.jpg"); %incarca Imagine.jpg in variabila I
imshow(I)%afiseaza imaginea salvata in variabila I
title('Imagine')
```

4.2 Reprezentarea canalelor RGB

```
I=imread("Imagine.jpg");
R=I(:,:,1); %Salveaza componenta 1 pentru fiecare element din matrice
G=I(:,:,2); %Salveaza componenta 2 pentru fiecare element din matrice
B=I(:,:,3); %Salveaza componenta 3 pentru fiecare element din matrice

%afisare
figure
subplot(2,2,1)
imshow(R)
title('Canal R')
subplot(2,2,2)
imshow(G)
title('Canal G')
subplot(2,2,3)
imshow(B)
title('Canal B')
subplot(2,2,4)
imshow(I)
title('Imagine color')
```

4.3 Histograme de imagini

Funcția folosită pentru realizarea reprezentărilor de tip histogramă de imagine în Octave/MATLAB este *imhist(imagine)*. **Atenție!** Această funcție face parte din pachetul **image**. Pentru a o utiliza trebuie ca pachetul respectiv să fie încărcat.

```
figure
subplot(2,2,1)
imhist(R)
title('Histograma Rosu')
subplot(2,2,2)
imhist(G)
title('Histograma Verde')
subplot(2,2,3)
imhist(B)
title('Histograma Albastru')
subplot(2,2,4)
imhist(I)
title('Histograma totala')
```

4.4 Convoluția bidimensională – filtre de imagine in scală de gri

În Octave/MATLAB convoluția bidimensională se realizează prin funcția *conv2(A,B,f)*, unde A și B sunt matricile incluse în operație, iar f reprezintă forma convoluției. Acest ultim argument poate lua valorile: *full* (valoarea implicită), *same* sau *valid*.

Atenție! Pentru ca operațiile de convoluție să se efectueze corect este necesar ca datele despre imagine să fie de tip double. La încărcarea unei imagini în Octave/MATLAB aceasta are valorile de tip uint8 (unsigned integer). Transformarea se face prin funcția *im2double()*.

```
I=imread("Imagine.jpg");
I=im2double(I);
```

Exemplificarea filtrelor de imagine este mai ușoară de realizat pe imagini în scală de gri. Transformarea unei imagini color într-una doar cu nuanțe de gri se face prin funcția *rgb2gray()*.

```
BW=rgb2gray(I);
```

```
N=1/9*ones(3,3); %nucleu normal

G3=1/16*[1 2 1;2 4 2;1 2 1]; %nucleu gaussian 3x3

G5=1/256*[1 4 6 4 1;4 16 24 16 4;6 24 36 24 6;4 16 24 16 4;1 4 6 4 1];%nucleu
gaussian 5x5


%convolutie

BWN=conv2(BW,N,'same');

BWG3=conv2(BW,G3,'same');

BWG5=conv2(BW,G5,'same');


%afisare

figure

subplot(2,2,1)

imshow(BW)

title('Imagine originala')

subplot(2,2,2)

imshow(BWN)

title('Estompare normala')

subplot(2,2,3)

imshow(BWG3)

title('Estompare gaussiana 3x3')

subplot(2,2,4)

imshow(BWG5)

title('Estompare gaussiana 5x5')
```

4.5 Filtre pentru imagini color

```
FR=[0 0.1 0;0.1 0.1 0.1;0 0.1 0];%nucleu pentru canalul R

FG=[0.5 0 0.5;0 -1 0;0.5 0 0.5];%nucleu pentru canalul G

FB=1/16*[1 2 1;2 4 2;1 2 1];%nucleu pentru canalul B

%Aplicarea filtrului

F(:,:,1)=conv2(I(:,:,1),FR,'same');

F(:,:,2)=conv2(I(:,:,2),FG,'same');

F(:,:,3)=conv2(I(:,:,3),FB,'same');


imshow(F)
```