UNIVERSITETI I PRISHTINËS
FAKULTETI I SHKENCAVE MATEMATIKO-NATYRORE
DEPARTAMENTI I MATEMATIKËS

LËNDA: Procesimi i imazheve
Projekti 3

Profesori:
Besnik Duriqi

Studentët:
Arita Osmani
Erëzana Berisha
Ermira Sekiraqa

16 Maj 2023, Prishtinë

1) Krijoni funksionin ‘zhurma’ i cili do të gjenerojë zhurmën me shpërndarje të Rayleigh, Eksponenciale, LogNormal, Salt&Pepper dhe Erlang. Funksioni duhet të merr këta parametra hyrës:

Zhurma(‘Shpërndarja’, x, y, z, k, s, a, b)

ku ‘Shpërndarja’ paraqet shpërndarjen me të cilën do të gjenerohet zhurma, x paraqet rreshtin prej ku do të filloj zhurma, y paraqet shtyllën prej ku do të filloj zhurma, z paraqet rreshtin prej ku do të përfundojë zhurma, k paraqet shtyllën prej ku do të përfundojë zhurma, numri i shtresave s ku s=1,2 ose 3, parametri a, parametri b (për parametrat a dhe b shikoni sllajdet).

- Për zgjidhjen e kësaj detyre, kemi implementuar funksionin Zhurma, i cili merr 8 parametra hyrës, siç është kërkuar në detyrë.

```
function [noise] = Zhurma(shperndarja, x, y, z, k, s, a, b)
```

Së pari, kontrollojmë nëse vlera e s-së është e ndryshme nga 1, 2 apo 3:

```
## Kontrollojme nese vlera e s eshte e ndryshme nga 1, 2 apo 3
if (s < 1 || s > 3)
  error('Vlera e s duhet te jete 1, 2 apo 3!');
endif
```

Pastaj, zgjidhim problemin në rastin kur a dhe b nuk jepen si parametra hyrës në funksion:

```
## Vlerat e paracaktuara te parametrave a dhe b
if nargin < 8
    a = 0;
    b = 0;
end
```

  - Funksioni *nargin* kthen numrin e parametrave hyrës të funksionit.
  - Nëse ky numër është më i vogël se 8, do të thotë se ato vlera (a dhe b) nuk janë dhënë si parametra hyrës në funksion, dhe rrjedhimisht iu vendosen vlerat e paracaktuara (0).

Caktojmë madhësinë e matricës së zhurmës:

```
m = z - x + 1;
n = k - y + 1;
```

Inicializojmë matricën zero të quajtur *zhurma*, me madhësi [m, n,s]:

```
## Krijimi i matrices se zhurmes
zhurma = zeros(m, n, s);
```

Në vazhdim, gjenerojmë zhurmën varësisht nga lloji i shpërndarjes:

```
## Gjenerimi i shurmes duke u bazuar ne tipin e shperndarjes
switch lower(shperndarja)
    case 'rayleigh'
        for i = 1:s
            zhurma(:,:,i) = a + b * random('Rayleigh', 1, [m, n]);
        end

    case 'exponential'
        for i = 1:s
            zhurma(:,:,i) = a + b * random('Exponential', 1, [m, n]);
        end

    case 'lognormal'
        for i = 1:s
            zhurma(:,:,i) = a + b * random('Lognormal', 0, 1, [m, n]);
        end

    case 'salt&pepper'
        for i = 1:s
            zhurma(:,:,i) = a + b * imzhurma(zeros(m, n), 'salt & pepper');
        end

    case 'erlang'
        for i = 1:s
            zhurma(:,:,i) = a + b * random('Erlang', 2, 1, [m, n]);
        end

    otherwise
        error('Shperndarja eshte e gabuar');
end
```

Për të ilustruar këtë funksion, do të paraqesim një imazh fillestar, e pastaj imazhin e zhurmshëm përmes funksionit të mësipërm me parametra të ndryshëm:

```
I = imread('C:\Users\Pulse Electronics\OneDrive\Desktop\Detyra3\lena.tif');
R = Zhurma("rayleigh", 1, 1, 256, 256, 3, 4, 7);
EXP = Zhurma("exponential", 1, 1, 256, 256, 3, 2, 2);
LOG = Zhurma("lognormal", 1, 1, 256, 256, 3, 2, 6);
SP = Zhurma("salt&pepper", 1, 1, 256, 256, 3, 6, 3);
ERL = Zhurma("erlang", 1, 1, 256, 256, 3, 7, 9);

subplot(2, 3, 1), imshow(I), title('Imazhi origjinal');
subplot(2, 3, 2), imshow(I + R), title('Rayleigh');
subplot(2, 3, 3), imshow(I + EXP), title('Exponential');
subplot(2, 3, 4), imshow(I + LOG), title('Lognormal');
subplot(2, 3, 5), imshow(I + SP), title('Salt&Pepper');
subplot(2, 3, 6), imshow(I + ERL), title('Erlang');
```

Fig. 1. Rezultatet e detyrës së parë.

2) Shtoni zhurmën periodike (mëposhtë) te imazhi 'lena.tif' dhe largojeni me anë të transformimeve Furie:

25*cos((pi*x)/4+(pi*y)/3)+ 10*sin((pi*x)/2+(pi*y)/9).

- Në këtë detyrë fillimisht shtojmë zhurmën ashtu siç kërkohet në detyrë, dhe pastaj e pastrojmë me anë të transformimeve Furie. Për të arritur këtë gjë së pari importojmë paketat që na nevojiten:

```
pkg load statistics
pkg load image
```

Pastaj e lexojmë foton në të cilën do të punojmë dhe e konvertojmë në double me anë të funksioneve imread dhe im2double:

```
img = imread('lena.tif');
img = im2double(img);
```

Gjenerohet një zhurmë e cila i shtohet imazhit origjinal. Përdoret funksioni Zhurma për të gjeneruar zhurmën Rayleigh. Funksioni përmban parametrat si: lloji i zhurmës, madhësia e imazhit dhe parametrat e zhurmës. Imazhi me zhurmë ruhet në një variabël *noisyimg*.

```
noisyimg = img + zhurma('rayleigh', 1, 1, size(img,1), size(img,2), 1, 0,
```

Zbatohet transformimi Fourier me anë të funksionit fft2 i cili merr një imazh hyrës (noisyimg në këtë rast) dhe llogarit FFT-në e tij 2-dimensionale. Në kontekstin e imazheve, 2-dimensional FFT llogarit përmbajtjen e frekuencës së një imazhi. Ai përfaqëson imazhin si një kombinim i frekuencave të ndryshme hapësinore, duke filluar nga frekuencat e ulëta deri te frekuencat e larta. Duke e aplikuar FFT në noisyimg, mund të analizojm përmbajtjen e frekuencës së imazhit dhe të kryejm operacione të tilla si filtrimi ose denoising në domenin e frekuencës.

```
F = fft2(noisyimg);
```

Më pas me funksionin meshgrid krijohet një rrjet koordinatash për frekuenca,

```
[X,Y] = meshgrid(1:size(noisyimg,2),1:size(noisyimg,1));
```

- size(noisyimg,2) kthen numrin e kolonave në matricën noisyimg, që përfaqëson gjerësinë e imazhit.
- size(noisyimg,1) kthen numrin e rreshtave në matricën noisyimg, që përfaqëson lartësinë e imazhit.

Funksioni meshgrid përdoret për të krijuar dy matrica, X dhe Y, që përfaqësojnë koordinatat X dhe Y të secilës pikë në rrjet. Matricat rezultuese X dhe Y kanë të njëjtat dimensione si matrica noisyimg.

Më pas bëhet llogaritja e frekuencave hapsinore me formulën që është kërkuar në detyrë

```
S = 25*cos((pi*X)/4+(pi*Y)/3) + 10*sin((pi*X)/2+(pi*Y)/9);
```

Bëhet ndërrimi i vendeve të frekuencave me anë të funksionit fftshift. Pra me anë të këtij funksioni pika me frekuencën zero do të pozicionohet në qendër në vend se të shkoj në skaj, kjo na jep vizualizim më të mirë dhe më të saktë.

```
S_shifted = fftshift(S);
```

Tash e shumëzojmë transformimin Furie me frekuencat e gjetura më lart

```
Multiplication = F .* S_shifted;
```

Dhe në fund bëjmë transformimin invers Furie të imazhit për ta kthyer nga domeni i frekuencës në atë hapsinorë siq ka qenë në fillim.

```
inv = ifft2(Multiplication);
```

Dhe shfaqja e imazheve në fund

```
subplot(1,2,1), imshow(img), title('Imazhi origjinal');
subplot(1,2,2), imshow(inv, []), title('Imazhi i krijuar');
```

Kur thirrim funksionin mund ta shohim vizualizisht se si ka ndikuar kodi në imazh.

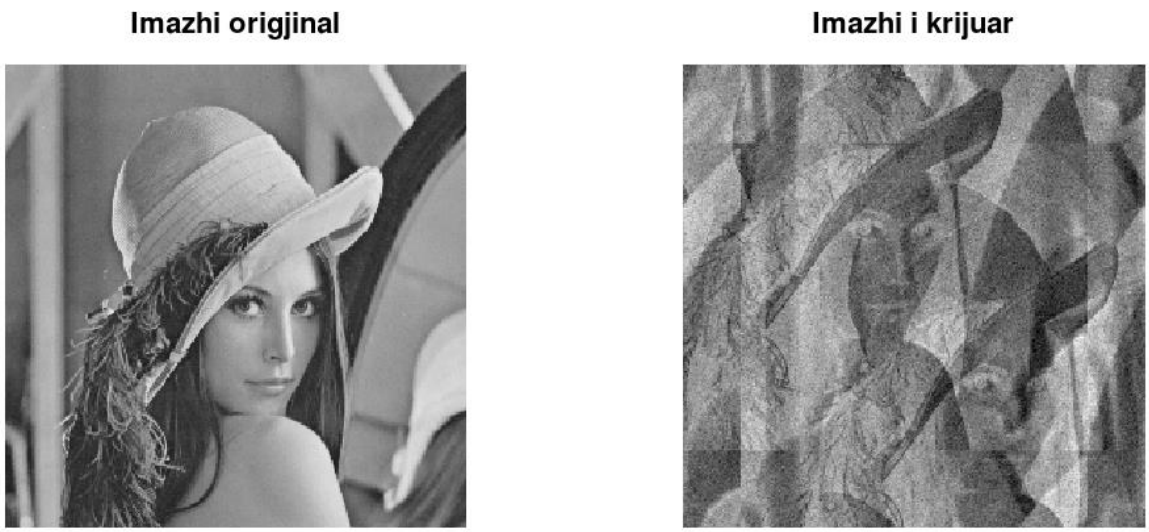

Fig. 2. Rezultatet e detyrës së dytë.

3) Krijoni filterin

$$H_B(u,v) = \frac{1}{1 + \left(\frac{D_0^2}{D_1(u,v)D_2(u,v)}\right)^n}$$

ku është qendra e njërës nga frekuencat që duhet të largohet (me gjithë të është edhe pika simetrike), $D_i(u,v) = \sqrt{\left(u - \frac{M}{2} + (-1)^i u_0\right)^2 + \left(v - \frac{N}{2} + (-1)^i v_0\right)^2}$, i = 1, 2 dhe M, N madhësia e imazhit.

- Për zgjidhjen e kësaj detyre kemi krijuar funksionin *filteri*, i cili merr 6 parametra hyrës dhe kthen vlerën H.

```
function H = filteri(M, N, u0, v0, d0, n)
```

Pjesa tjetër e kodit krijon një rrjetë frekuencesh duke përdorur funksionin meshgrid. Funksioni meshgrid krijon matricat U dhe V që përfaqësojnë koordinatat e rrjetit të frekuencave.

```
## Creating the grid for frequencies
[U, V] = meshgrid(1:N, 1:M);
```

Pastaj, kodi llogarit distancat nga qendra për çdo frekuencë. Distancat D1 dhe D2 llogariten sipas formulës së specifikuar në detyrë.

```
## Calculating distances from the center for each frequency
D1 = sqrt((U - (M/2+u0)).^2 + (V - (N/2+v0)).^2);
D2 = sqrt((U - (M/2-u0)).^2 + (V - (N/2-v0)).^2);
```

Pjesa tjetër e kodit përcakton funksionin e filtrimit:

```
## Filter function
H = 1 ./ (1 + (d0 ./ (D1 .* D2)).^n);
```

Pas definimit të funksionit, marrim një imazh:

```
I = imread('C:\Users\Pulse Electronics\OneDrive\Desktop\Detyra3\lena.tif
I = im2double(I);
```

Aplikojmë transformimin Furie dhe filterin *filteri* në imazh:

```
F = fftshift(fft2(I));

H = filteri(size(I,1), size(I,2), 50, 70, 20, 3);

G = F .* H;
```

Në fund, aplikojmë transformimin invers Furie në imazh dhe shfaqim rezultatet:

```
filteredI = real(ifft2(ifftshift(G)));

subplot(1, 2, 1), imshow(I), title('Imazhi origjinal');
subplot(1, 2, 2), imshow(filteredI), title('Imazhi i filtruar');
```

Fig. 3. Rezultatet e detyrës së tretë.

Në shikim të parë nuk është se vërejmë ndonjë dallim mes këtyre dy imazheve, mirëpo padyshim që ka dallim mes tyre. Dallimet mund të vërehen më mirë në figurën e mëposhtme ku kemi paraqitur vlerat e dy imazheve:

I [256x256 double]

	1	2	3	4	5	6
1	0.64706	0.64706	0.64706	0.65098	0.65882	0.64706
2	0.64706	0.64706	0.64706	0.65098	0.65882	0.64706
3	0.64706	0.64706	0.64706	0.65098	0.65882	0.64706
4	0.63922	0.64706	0.63922	0.63529	0.63529	0.63137
5	0.61961	0.63137	0.63529	0.62745	0.65098	0.63529
6	0.62353	0.61961	0.62353	0.61961	0.63137	0.62745
7	0.63137	0.62745	0.62353	0.63529	0.63922	0.63529
8	0.63137	0.63529	0.61569	0.62745	0.62745	0.60392

filteredI [256x256 double]

	1	2	3	4	5	6
1	0.64686	0.64671	0.64702	0.6513	0.65909	0.64692
2	0.64679	0.64719	0.64742	0.6511	0.65854	0.64674
3	0.64733	0.64737	0.647	0.65062	0.65863	0.64729
4	0.63941	0.64684	0.63886	0.63527	0.63564	0.63163
5	0.61928	0.63111	0.63545	0.62782	0.65108	0.63498
6	0.62342	0.61991	0.62384	0.61952	0.631	0.62727
7	0.63174	0.62763	0.62328	0.63494	0.63922	0.63566
8	0.63138	0.63494	0.61544	0.62764	0.62783	0.604

Fig. 4. Vlerat e imazhit origjinal dhe atij të filtuar në detyrën e tretë.

4) Është dhënë regjioni binar (Figura 5) dhe regjioni binar pas veprimit me element strukturor në regjionin binar origjinal (Figura 6). Gjeni elementin strukturor dhe operacionet nga matematika morfologjike që do të japin rezultatin nga Figura 6.

Fig. 5. Regjioni binar origjinal.

Fig. 6. Regjioni pas veprimit me element strukturor në regjionin binar origjinal.

- Ideja e zgjidhjes së kësaj detyre është që të fillohet nga imazhi origjinal dhe me disa operacione matematike që veprojnë në të, të kthehet në imazhin e kërkuar në detyrë. Prandaj fillimisht marrim imazhin origjinal:

Konvertojmë imazhin në grayscale format:

```
original_img = rgb2gray(original_img);
```

Kthejmë imazhin në imazh binar:

```
original_img = im2bw(original_img, 0.5);
```

Funksioni im2bw konverton imazhin hyrës në një imazh binar. Pikseleve me intensitet më të madh ose të barabartë me vlerën e pragut (0.5 në rastin tonë) u caktohet vlera 1 (e bardhë), ndërsa pikselëve me intensitet më të ulët se vlera e pragut u caktohet vlera 0 (e zezë).
Më pas krijojmë një element strukturor në formën e një katrori me një madhësi prej 26 pikselësh në secilën anë.

```
se = strel('square',26);
```

Dhe tani aplikojmë erosionin me anë të funksionit imerode që merr si parametër imazhin dhe elementin e krijuar me lart.

```
output_img = imerode(original_img, se);
```

Në fund shfaqim dy imazhet, imazhin fillestar dhe imazhin e krijuar.

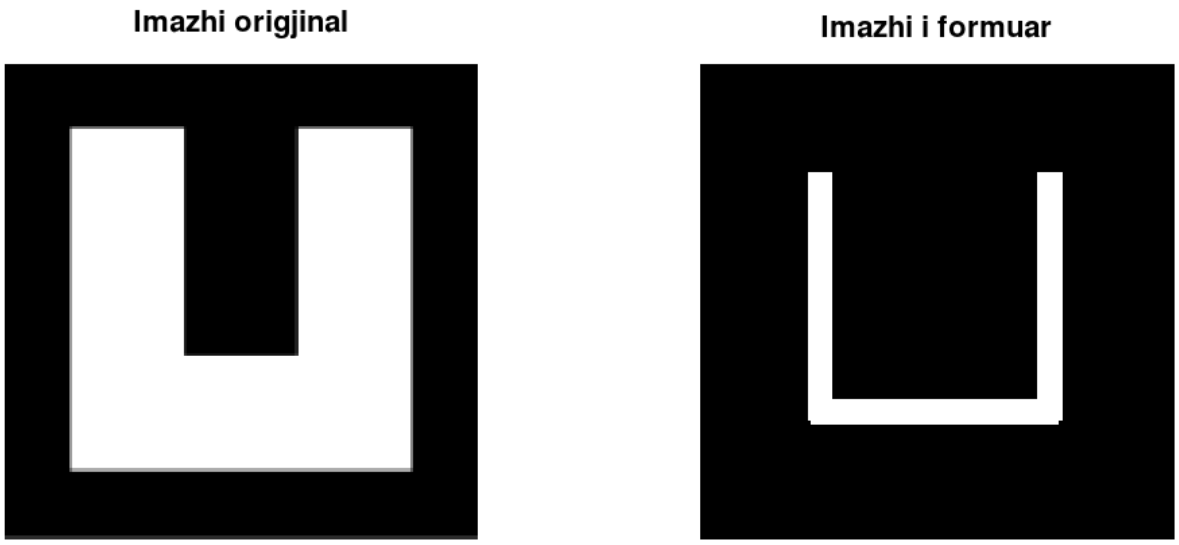

Fig. 7. Rezultatet e detyrës së katërt.