UNIVERSITETI I PRISHTINËS

FAKULTETI I SHKENCAVE MATEMATIKE-NATYRORE

DEPARTAMENTI I MATEMATIKËS

****

*LËNDA: Procesim i imazheve*

Profesori: Studentët:

Besnik Duriqi Ermira Maliqi

Festina Klinaku

Jusufe Selimi

16 Maj 2023, Prishtinё

### **Kërkesat e detyrave**

**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 te filloj zhurma, y paraqet shtyllën prej ku do të filloj zhurma, z paraqet rreshtin prej ku do te 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). (3 pikë)

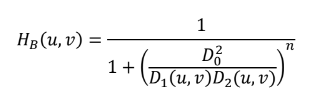
**2.** Shtoni zhurmën periodike (mëposhtë) te imazhi ‘lena.tif’ dhe largojeni me anë të transformimeve Furie? (4 pikë)

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



Fig.1. Imazhi ‘lena.tif’

**3.** Krijoni filterin



ku D0 = (u0, v0) është qendra e njërës nga frekuencat që duhet të largohet (me gjithë të është edhe



pika simetrike), D1 (u, v) = , i = 1, 2 dhe M, N

madhësia e imazhit. (4 pikë)

**4.** Është dhënë regjioni binar (Figura 2) dhe regjioni binar pas veprimit me element strukturor

në regjionin binar origjinal (Figura 3). Gjeni elementin strukturor dhe operacionet nga matematika morfologjike që do të japin rezultatin nga Figura 3. (4 pikë)

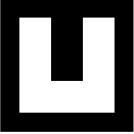


Fig.2. Regjioni binar origjinal

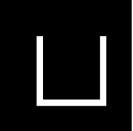


Fig.3. Regjioni pas veprimit me element strukturor në regjionin binar origjinal

### **Zgjidhja e detyrave**

#### **Përshkrimi i kodit për detyrën 1:** Kodi për funksionin ‘zhurma’ është krijuar në një file të emërtuar ‘zhurma.m’ përmes MatLab.

function [noise] = zhurma(distribution, x, y, z, k, s, a, b)

% Vlerat e paracaktuara te parametrave a dhe b

if nargin < 8

a = 0;

b = 0;

end

% Krijimi i matrices se zhurmes

noise = zeros(z - x + 1, k - y + 1, s);

% Gjenerimi i shurmes duke u bazuar ne tipin e shperndarjes

switch lower(distribution)

case 'rayleigh'

for i = 1:s

noise(:,:,i) = a + b\*random('Rayleigh', 1, [z - x + 1, k - y + 1]);

end

case 'exponential'

for i = 1:s

noise(:,:,i) = a + b\*random('Exponential', 1, [z - x + 1, k - y + 1]);

end

case 'lognormal'

for i = 1:s

noise(:,:,i) = a + b\*random('Lognormal', 0, 1, [z - x + 1, k - y + 1]);

end

case 'salt&pepper'

for i = 1:s

noise(:,:,i) = a + b\*imnoise(zeros(z - x + 1, k - y + 1), 'salt & pepper');

end

case 'erlang'

for i = 1:s

noise(:,:,i) = a + b\*random('Erlang', 2, 1, [z - x + 1, k - y + 1]);

end

otherwise

error('Invalid distribution type');

end

end

#### **Përshkrimi i kodit për detyrën 2:** Kodi për detyrën e dytë përmban pjesën e aplikimin të funksionit zhurma në imazhin ‘lena.tif’ të paraqitur në figurën 4 dhe të largimit përmes tranformimeve Furie. I gjithë kodi është ruajtur në një file të emërtuar ‘furie.m’.

% Pastrimi i workspace dhe dritares komanduese

clc

clear all

close all

% Importimi i paketave te nevojshme

pkg load image

pkg load statistics

% Aplikimi i zhurmes ne imazhin 'lena.tif'

img = imread('lena.tif')

img = im2double(img)

noisy\_img = img + zhurma('rayleigh', 1, 1, size(img,1), size(img,2), 1, 0, 0.1);

figure;

subplot(1,2,1); imshow(img); title('Original Image');

subplot(1,2,2); imshow(noisy\_img); title('Noisy Image');

Me ekzekutimin e kodit të mësipërm fitojmë rezultatet e mëposhtme:

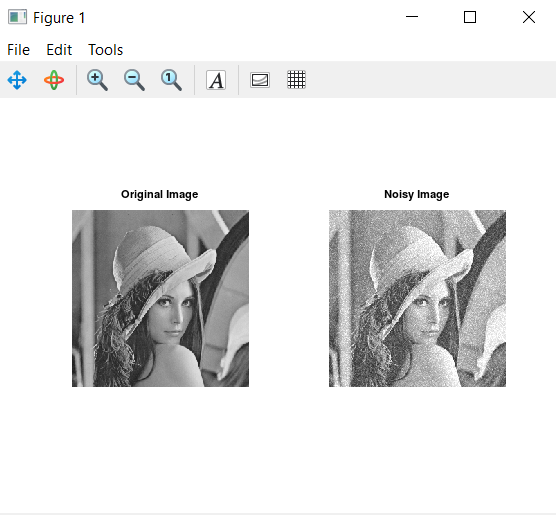


Fig.4. Aplikimi i zhurmës në imazhit ‘lena.tif’

% Perdorimi i transformimit Furie ne imazhit e zhurmshem

F = fft2(noisy\_img);

% Krijimi i grid per frekuencat

[X,Y] = meshgrid(1:size(noisy\_img,2),1:size(noisy\_img,1));

% Llogaritja e frekuencave spatiale permes formules

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

% Nderrimi i vendeve te frekuencave spatiale

H\_shifted = fftshift(H);

% Shumezimi i tranformimit Furie te imazhit te zhurmshem me frekuencen spatiale

G = F \* H\_shifted;

% Tranformimi invers Furie

g = ifft2(G);

% Shfaqja e imazhit pas tranformimit

imshow(g, [])

Me ekzekutimin e kodit të mësipërm fitojmë rezultatin e mëposhtëm në të cilin mund të shihet se funksioni i përzgjedhur nuk është më i miri për largimin e zhurmës nga ky imazh.

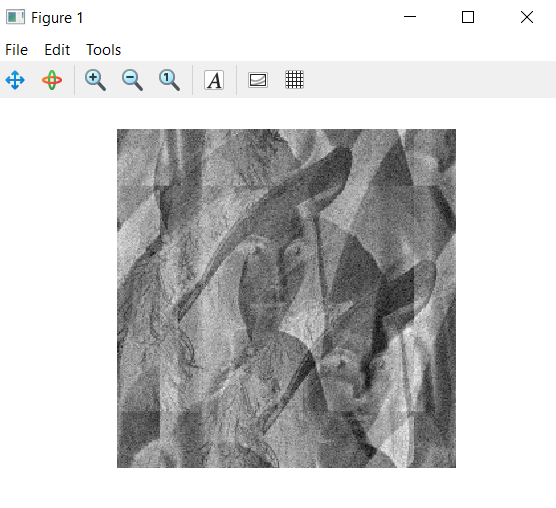


Fig.5. Aplikimi tranformimeve Furie në imazhin e zhurmshëm

#### **Përshkrimi i kodit për detyrën 3:** Kodi per krijimin dhe aplikimin e filtrit është ruajtur në një file me emërtimin ‘myfilter.m’ ku funksioni myfilter e paraqet pjesën për krijimin e filtrit ndërsa pjesa tjetër aplikimin e tij.

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

% Krijimi i grid per frekuencat

[U, V] = meshgrid(1:N, 1:M);

% Llogaritja e distances nga qendra per secilen frekuence

D1 = sqrt((U - (M/2+u0)).^2 + (V - (N/2+v0)).^2);

D2 = sqrt((U - (M/2-u0)).^2 + (V - (N/2-v0)).^2);

% Funksioni per filtrim

H = 1 ./ (1 + (d0 ./ (D1 .\* D2)).^n);

end

img = imread('lena.tif');

img = im2double(img);

% Transformimi Furie

F = fftshift(fft2(img));

% Definimi i parametrave

u0 = 30;

v0 = 20;

d0 = 50;

n = 4;

% Perdorimi i funksionit per krijimit e filtrit

H = myfilter(size(img,1), size(img,2), u0, v0, d0, n);

% Aplikimi i filtrit ne imazh

G = F .\* H;

% Inversi i transformimit Furie

g = real(ifft2(ifftshift(G)));

% Shfaqja e imazhit

imshowpair(img, g, 'montage');

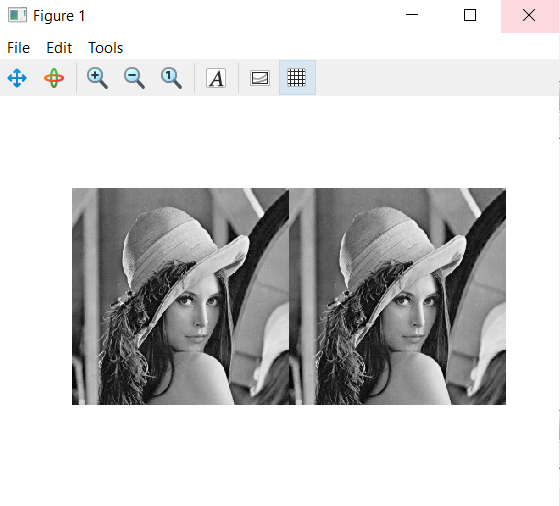


Fig.6. Aplikimi i filtrit në imazhin e fituar nga transformimi Furie

Që të dallohet më mirë ndryshimi kemi shfaqur edhe ndryshimin e vlerave të elementeve të matricës së imazheve.



#### **Përshkrimi i kodit për detyrën 4:** Kodi për detyrën 4 është ruajtur në një file me emërtimin ‘erosion.m’ ku kemi provuar të performojmë erozionin me disa elemente të ndryshme strukturore dhe në kodin e mëposhtëm kemi paraqitur elementin strukturor nga i cili kemi marrë rezultatin më të saktë.

% Load the original binary image and the output binary image

original\_bw = imread('original\_image.png');

eroded\_bw = imread('eroded\_image.png');

% Convert to grayscale (if necessary)

if size(original\_bw, 3) == 3

original\_bw = rgb2gray(original\_bw);

end

if size(eroded\_bw, 3) == 3

eroded\_bw = rgb2gray(eroded\_bw);

end

% Convert to binary images using Otsu's threshold

original\_bw = im2bw(original\_bw, 0.5);

eroded\_bw = im2bw(eroded\_bw, 0.5);

% Define a structuring element to try

se = strel('square',26);

% Perform erosion using the structuring element

output\_bw = imerode(original\_bw, se);

% Show image

figure, imshow(output\_bw);

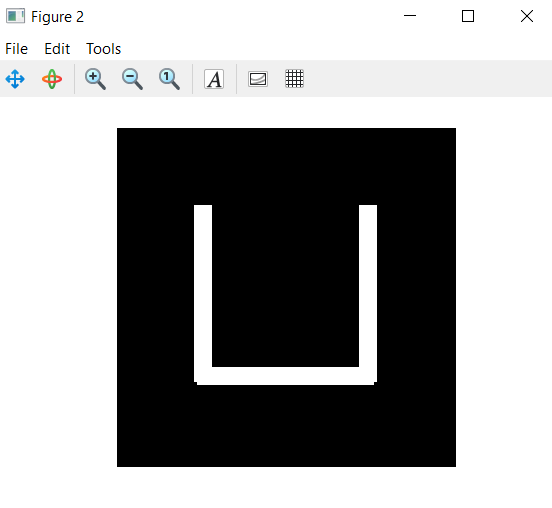
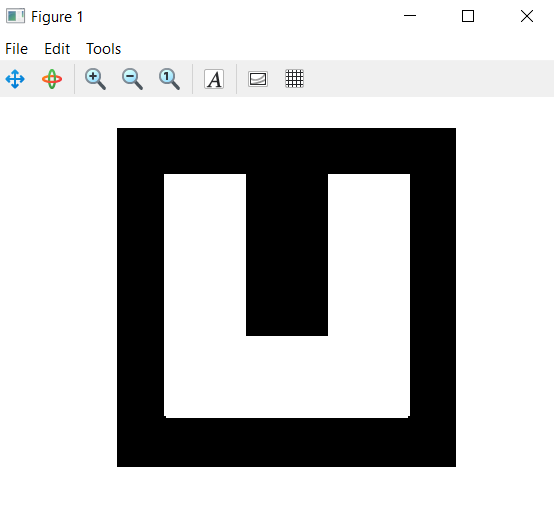


Fig.7. Rezultati nga erozioni me elementin strukturor të përzgjedhur

### **Referencat**

[1] Woods, R. E., Gonzalez, R. C. (2018). Digital Image Processing. United Kingdom: Pearson.

[2] https://www.mathworks.com/?s\_tid=gn\_logo