Video Signals

Date: 20 July 2026

Ex.1.[10 pts] An 8×8 grayscale image, with double-precision intensity values between 0 and 1, has the following 2D Fourier transform.

32	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	32	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

[6pts] Describe the steps required to obtain the original image.
What will be the original image?

[4 pts] What would be the 2D Fourier transform of the negative image, i.e., where each pixel value is given by 1 minus the current value of the same pixel?

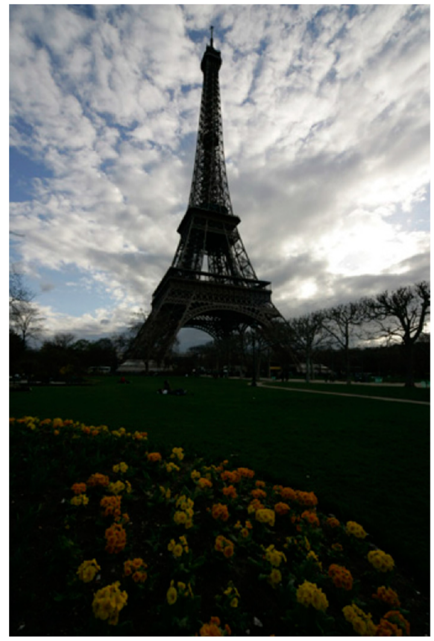
Ex.2. [11 pts] Given a 4×4 image (**I**), we aim to extract its edges using the Laplacian operator. Describe the Laplacian operator and explain the procedure used to detect the image contours after filtering the image with this operator.

$$I = \begin{bmatrix} 7 & 8 & 10 & 10 \\ 8 & 9 & 10 & 5 \\ 5 & 3 & 2 & 8 \\ 3 & 2 & 8 & 9 \end{bmatrix}$$

Provide an image with the approximate edges position.

Ex. 3 is overleaf

Es.3. [12 pts to be solved writing MATLAB code on the paper]



Images acquired with a camera may suffer from non-uniform illumination. Some regions may appear darker because of shadows, while others may be brighter due to uneven lighting conditions. This effect can reduce the visual quality of the image and make further image processing tasks more difficult.

In this exercise, you will implement a simple color illumination correction algorithm. The idea is to work in the HSV color space and correct only the Value channel, which represents image brightness. The illumination field is estimated by applying a large Gaussian filter to the Value channel. The corrected brightness is then combined with the original brightness to obtain a natural-looking result.

Starting from the input image photo.png, write a MATLAB script able to perform the following steps:

a) Read the image photo.png, convert it to double representation, and display it.

b) Convert the RGB image to the HSV color space.

c) Extract the Hue, Saturation, and Value channels.

d) Estimate the illumination field by applying a Gaussian filter to the Value channel. Use a kernel size of 101x101 and standard deviation equal to 30.

e) Normalize the illumination field by dividing it by its average value.

f) Correct the Value channel by dividing it by the normalized illumination field. Use a small constant to avoid numerical problems in the division.

g) Clip the corrected Value channel to the range [0, 1].

h) Combine the corrected Value channel with the original Value channel using a weighted average. Use equal weights for the two channels.

i) Replace the original Value channel with the new one, convert the image back to RGB, and display the final corrected image.

List of possible Matlab functions

```
figure
im2double
imread
rgb2gray
imcrop
imfilter
imrotate
imopen
rgb2hvs
histeq
hist
imshow
fspecial
imerase
strel
hsv2rgb
imbinarize
```

Solutions

Es.1

Since we have a component at the continuous component and one at the Nyquist frequency the inverse Discrete Fourier Transform:

1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1

In order to change such an image to its negative we can reverse the phase of the Nyquist component. The 2D Discrete Fourier Trasform becomes:

32	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	-32	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

And the image will become:

0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	1	0

Es.2

Given the laplacian:

0	1	0
1	-4	1
0	1	0

We get the following filtered image

$$I_{filtered} = \begin{bmatrix} -12 & -6 & -12 & -25 \\ -11 & -7 & -14 & 8 \\ -6 & 6 & 21 & -16 \\ -5 & 6 & -19 & -20 \end{bmatrix}$$

The edge will be at zeros of the filtered image, so, approximatively:

-12	-6	-12	-25
-11	-7	-14	8
-6	6	21	-16
-5	6	-19	-20

Es.3

```
clc
close all
clear all

% a) Read the image
```

```
I = im2double(imread('photo.png'));

figure
imshow(I)
title('Original image')

% b) Convert to HSV
I_hsv = rgb2hsv(I);

% c) Extract HSV channels
H = I_hsv(:,:,1);
S = I_hsv(:,:,2);
V = I_hsv(:,:,3);

% d) Estimate the illumination field from the Value channel
G = fspecial('gaussian', [101 101], 30);
illumination = imfilter(V, G, 'conv', 'same', 'symmetric');

% e) Normalize the illumination field
illumination = illumination / mean(illumination(:));

% f) Correct the Value channel
epsilon = 1e-6;
V_corrected = V ./ (illumination + epsilon);

% g) Clip values to the range [0, 1]
V_corrected = min(V_corrected, 1);
V_corrected = max(V_corrected, 0);

% h) Combine corrected and original Value channels
alpha = 0.5;
V_final = alpha * V_corrected + (1 - alpha) * V;

% i) Replace the Value channel and convert back to RGB
I_hsv_corrected = I_hsv;
I_hsv_corrected(:,:,3) = V_final;

I_corrected = hsv2rgb(I_hsv_corrected);

figure
imshow(I_corrected)
title('Color illumination corrected image')
```