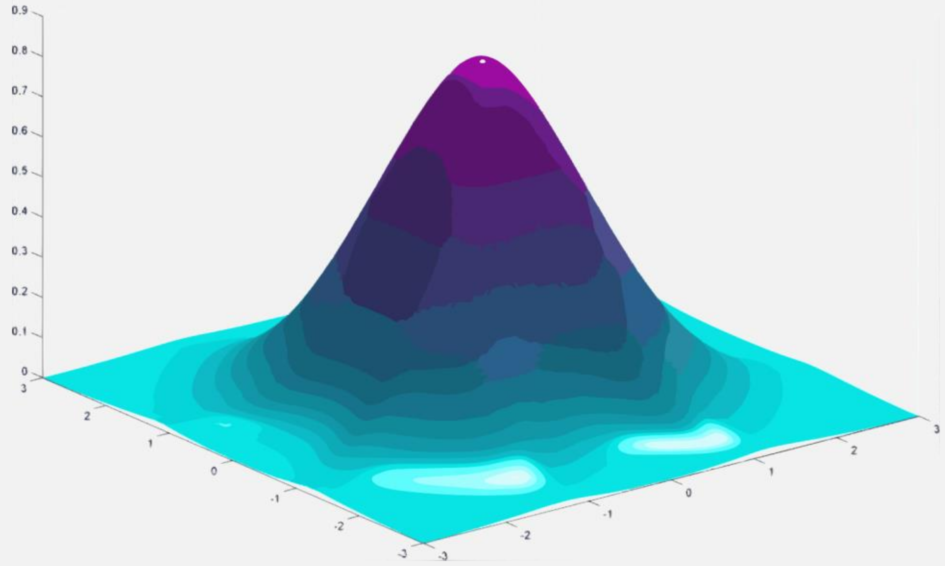
ΨΗΦΙΑΚΗ ΕΠΕΞΕΡΓΑΣΙΑ ΕΙΚΟΝΑΣ

Αναφορά 3ης Εργαστηριακής Άσκησης

«2-D ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER ΣΕ ΕΙΚΟΝΑ»



Καραμαΐλής Παντελής 2016030040
Βλάχος Κωνσταντίνος 2016030042
Γαλάνης Μιχάλης 2016030036

ΠΕΡΙΕΧΟΜΕΝΑ

Οι σύνδεσμοι για τις παρακάτω ενότητες είναι διαδραστικοί.
Πιέστε πάνω στο επιθυμητό τμήμα της άσκησης για τη μετάβαση σε αυτό.

ΕΙΣΑΓΩΓΗ

Σκοπός Εργαστηρίου

Σκοπός της άσκησης αυτής είναι η ανάλυση του δυσδιάστατου διακριτού μετασχηματισμού Fourier μιας εικόνας. Μεταβάλλοντας διάφορα χαρακτηριστικά της, μελετάμε πως επηρεάζεται ο μετασχηματισμός της και επεξηγούμε τα αποτελέσματά του βασιζόμενοι στη θεωρία. Για την αναφορά αυτή, συγχωνεύουμε τη θεωρητική ανάλυση με την υλοποίηση σε μια ενότητα για λόγους απλότητας.

1 – M.F ΕΙΚΟΝΑΣ

Στο πρώτο ερώτημα μας ζητήθηκε να διαβάσουμε μια εικόνα και να απεικονίσουμε το μέτρο του μετασχηματισμού Fourier της. Στη συνέχεια έπρεπε να δείξουμε το μετασχηματισμό μετατοπισμένο στο κέντρο με κατάλληλη τιμή εντάσεων σε grayscale αλλά και σε color-mapped μορφή.

Θεωρητική Ανάλυση – Υλοποίηση

Ο μετασχηματισμός Fourier συνιστά δυνατό εργαλείο για την επεξεργασία μιας εικόνας. Την αποσυνθέτει σε ημίτονα και συνημίτονα και η εφαρμογή του την παρουσιάζει στο πεδίο της συχνότητας. Ο διακριτός μετασχηματισμός Fourier μιας εικόνας ορίζεται ως:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}$$

Στη matlab, εφαρμόζουμε μετασχηματισμό με τη συνάρτηση **`fft2()`** ενώ για να τον μετατοπίσουμε στο κέντρο χρησιμοποιούμε την **`fftshift()`** .

Για να απεικονίσουμε ικανοποιητικά τον μετασχηματισμό ως εικόνα, δηλαδή να έχει ως τιμές εντάσεων μεταξύ 0 και 255, χρησιμοποιούμε το μετασχηματισμό:

$$D(u,v) = c\log[1 + |F(u,v)|]$$

Όπου c μια σταθερά που λειτουργεί ως ευαισθησία του μετασχηματισμού. Βρέθηκε με πειραματισμό ότι για απεικόνιση σε κλίμακα αποχρώσεων του γκρι χρειάστηκε $c = 0.04$. Για την απεικόνιση σε color map χρησιμοποιήθηκε $c = 4$ με έτοιμη χρωματική παλέτα **`jet()`** από τη matlab που περιέχει όλα τα χρώματα των συνδυασμών κόκκινου, πράσινου και μπλε καναλιού.

Τέλος, συγκρίνουμε τα αποτελέσματα με χρήση **`figure`** και **`subplot`**.

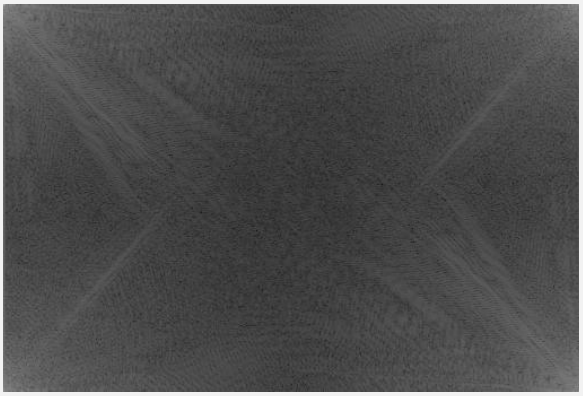
Μετάβαση στον κώδικα MATLAB

Αποτελέσματα & Συμπεράσματα

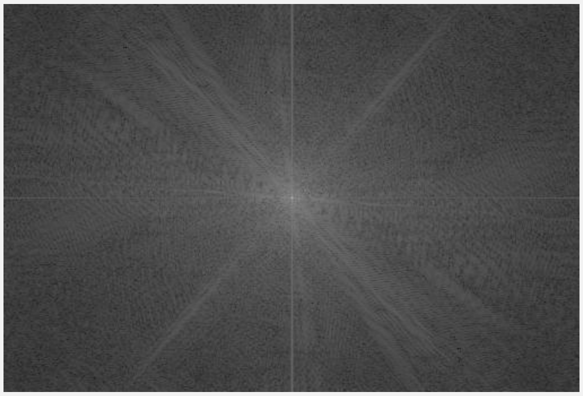
Παρακάτω παραθέτουμε τις εικόνες και τους μετασχηματισμούς που προέκυψαν από τα ζητούμενα της εκφώνησης:

ΑΡΧΙΚΗ ΕΙΚΟΝΑ

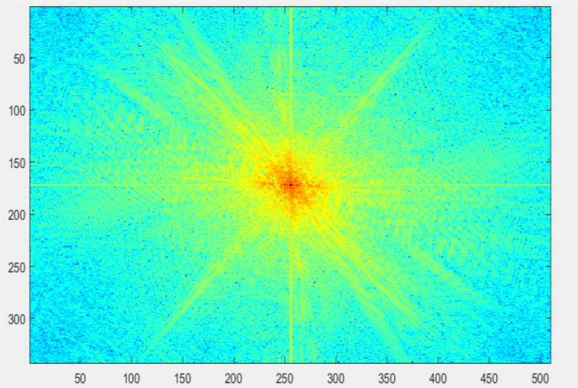
ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ ΕΙΚΟΝΑΣ ΧΩΡΙΣ ΜΕΤΑΤΟΠΙΣΗ



ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ ΕΙΚΟΝΑΣ ΜΕ ΜΕΤΑΤΟΠΙΣΗ (GRAYSCALE)



ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ ΕΙΚΟΝΑΣ ΜΕ ΜΕΤΑΤΟΠΙΣΗ (COLOR MAP)



Τα αποτελέσματα είναι αναμενόμενα. Παρατηρούμε ότι οι grayscale μετασχηματισμοί της εικόνας παίρνουν τιμές εντάσεων από 0 έως 255, ενώ ο μετασχηματισμός της εικόνας color map απεικονίζει τις τιμές χρωματική παλέτα jet. Ακόμα, με τη μετατόπιση του μετασχηματισμού της εικόνας, βλέπουμε ότι η πληροφορία της συσσωρεύεται στο κέντρο της.

2 – M.F ΕΙΚΟΝΑΣ ΜΕ ΜΟΤΙΒΑ

Στη δεύτερη άσκηση, εφαρμόζουμε μετασχηματισμό Fourier σε 2 εικόνες που για χαρακτηριστικά έχουν ότι η μία αποτελείται από πολλές οριζόντιες ακμές, ενώ η άλλη από πολλές κάθετες ακμές.

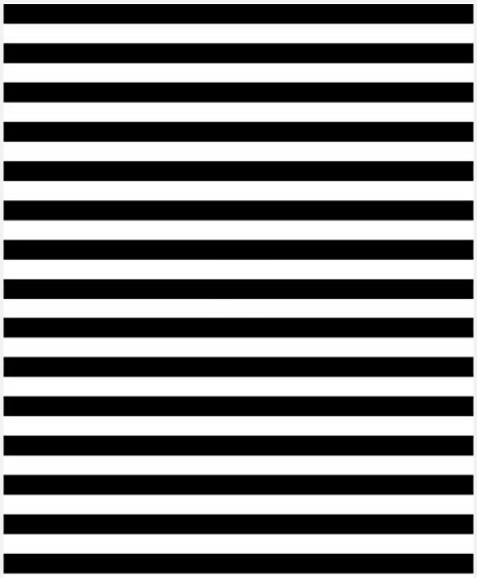
Θεωρητική Ανάλυση – Υλοποίηση

Διαβάζουμε ως συνήθως τις 2 εικόνες. Εφαρμόζουμε μετασχηματισμό Fourier όπως στη προηγούμενη άσκηση και με κατάλληλες τιμές c απεικονίζουμε και συγκρίνουμε τους μετασχηματισμούς σε δυο ξεχωριστά figures.

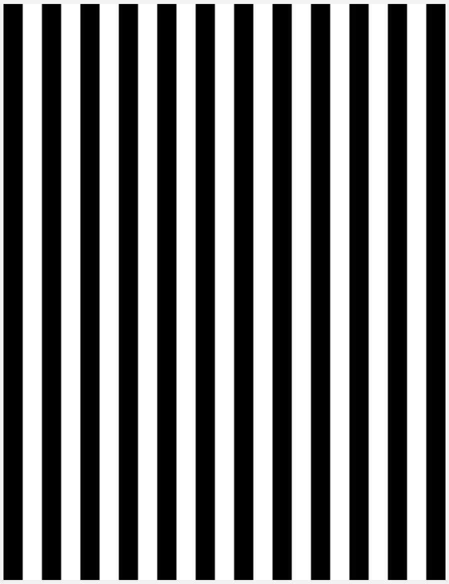
Μετάβαση στον κώδικα MATLAB

Αποτελέσματα & Συμπεράσματα

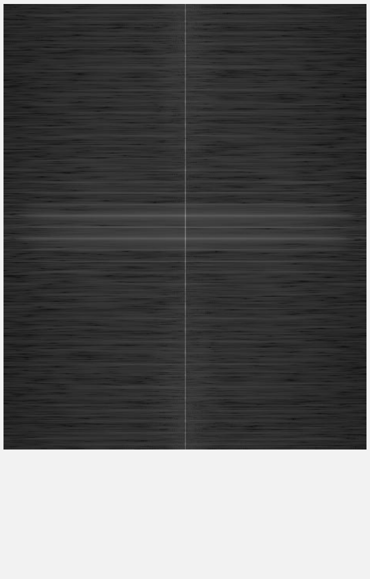
**ΑΡΧΙΚΗ ΕΙΚΟΝΑ
ΟΡΙΖΟΝΤΙΕΣ ΓΡΑΜΜΕΣ**



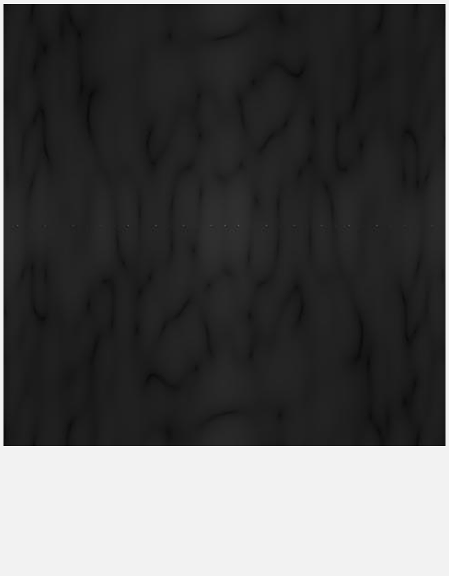
**ΑΡΧΙΚΗ ΕΙΚΟΝΑ
ΚΑΘΕΤΕΣ ΓΡΑΜΜΕΣ**



**ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER
ΟΡΙΖΟΝΤΙΕΣ ΓΡΑΜΜΕΣ**



**ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER
ΚΑΘΕΤΕΣ ΓΡΑΜΜΕΣ**



Παρατηρούμε ότι για την εικόνα με οριζόντιες ακμές ο μετασχηματισμός Fourier της περιλαμβάνει μια κάθετη γραμμή στο φάσμα της, ενώ για αυτή με κάθετες ακμές, ο μετασχηματισμός Fourier έχει φωτεινά σημεία που σχηματίζουν οριζόντια γραμμή. Και στους δύο μετασχηματισμούς παρατηρείται «θόρυβος» στο background της εικόνας που οφείλεται σε διάφορους παράγοντες.

3 – M.F ΤΕΧΝΙΤΗΣ ΕΙΚΟΝΑΣ

Η τρίτη άσκηση αποσκοπεί στη μελέτη του μετασχηματισμού Fourier στη περίπτωση που δημιουργούμε δύο νέες εικόνες 256 x 256 οι οποίες έχουν μαύρο χρώμα παντού εκτός από ένα τετράγωνο στο κέντρο της εικόνας που θα είναι λευκό. Τα μεγέθη των τετραγώνων είναι 10 x 10 και 30 x 30 αντίστοιχα. Εφαρμόζουμε μετασχηματισμό Fourier και συγκρίνουμε τα αποτελέσματά τους.

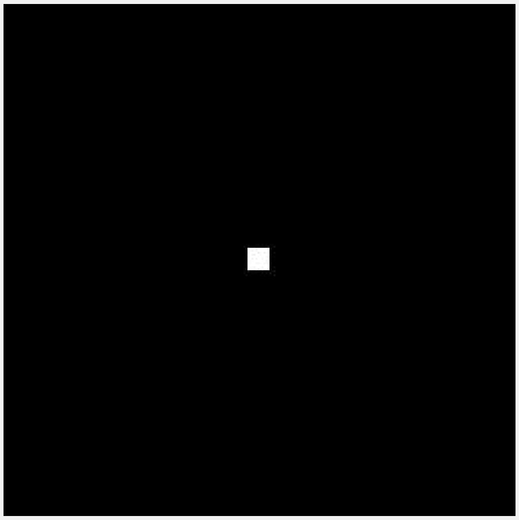
Θεωρητική Ανάλυση – Υλοποίηση

Ο κώδικας έχει γίνει παραμετροποιημένος. Αρχικά διατηρούμε σε 2 μεταβλητές την ανάλυση της επιθυμητής εικόνας και έναν πίνακα με τις τιμές των διαμέτρων των τετραγώνων. Στην περίπτωσή μας έχουμε **`diameters = [10, 30]`** και **`image_resolution = 256`**. Για κάθε τιμή του πίνακα **`diameters`**, δημιουργούμε έναν δισδιάστατο πίνακα **`image`** αρχικοποιημένος με μηδενικά (που υποδηλώνουν το μαύρο) και βρίσκουμε το κατάλληλο χώρο που θα αλλάξουμε τις τιμές των pixel σε λευκό. Για κάθε pixel της εικόνας, ελέγχουμε εάν βρίσκεται σε αυτή τη περιοχή, και αλλάζουμε την τιμή του pixel από 0 σε 255. Μετασχηματίζουμε με Fourier την τελική εικόνα και τις παρουσιάζουμε όλες μαζί σε κοινό figure.

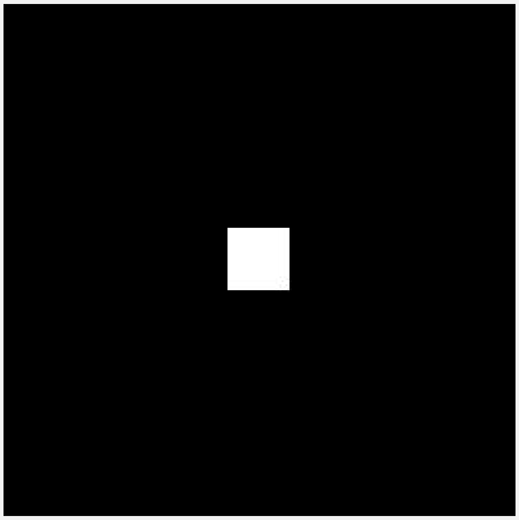
Μετάβαση στον κώδικα MATLAB

Αποτελέσματα & Συμπεράσματα

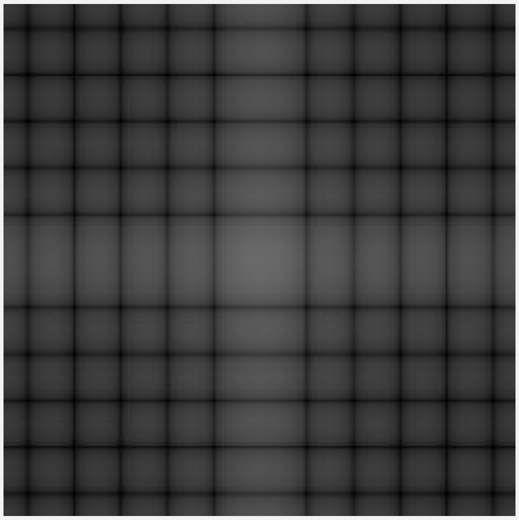
ΑΡΧΙΚΗ ΕΙΚΟΝΑ (10 x 10)



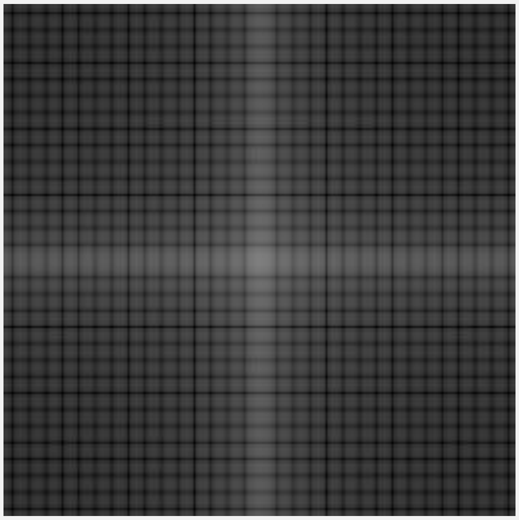
ΑΡΧΙΚΗ ΕΙΚΟΝΑ (30 x 30)



ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER (10 x 10)



ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER (30 x 30)



Παρατηρούμε ότι όσο μεγαλώνει το μέγεθος του κεντρικού τετραγώνου, αυξάνεται το πλήθος των τετραγώνων στην απεικόνιση του μετασχηματισμού και μειώνεται το εμβαδόν τους. Οι επαναλήψεις του δηλαδή είναι περισσότερες και μικρότερες σε φάσμα. Το θεωρητικό υπόβαθρο του αποτελέσματος αυτού στηρίζεται στην ιδιότητα ότι ό πολλαπλασιασμός στο χρόνο ισοδυναμεί με διαίρεση στη συχνότητα.

4 – M.F ΠΕΡΙΣΤΡ. ΤΕΧΝΙΤΗΣ ΕΙΚΟΝΑΣ

Στη τέταρτη άσκηση μας ζητείται να επαναλάβουμε τη προηγούμενη διαδικασία αλλά πριν το μετασχηματισμό Fourier να περιστραφεί το κεντρικό παράθυρο κατά 45°.

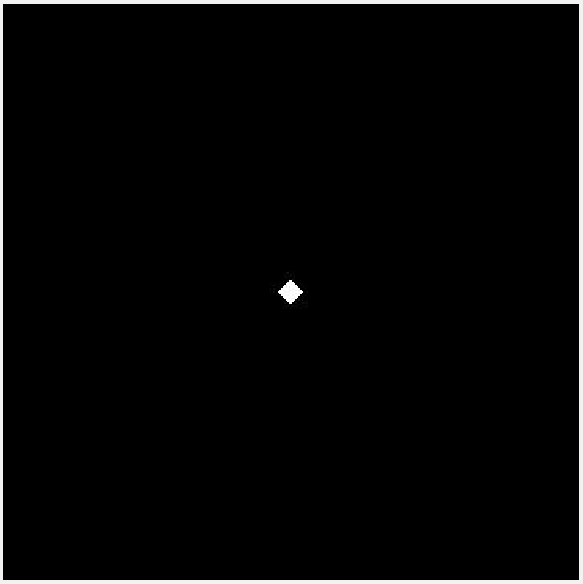
Θεωρητική Ανάλυση – Υλοποίηση

Η υλοποίηση είναι ακριβώς η ίδια με προηγουμένως. Η μόνη διαφορά είναι ότι με τη δημιουργία της εικόνας, χρησιμοποιούμε τη συνάρτηση **`imrotate()`** για να περιστρέψουμε την εικόνα πριν το μετασχηματισμό.

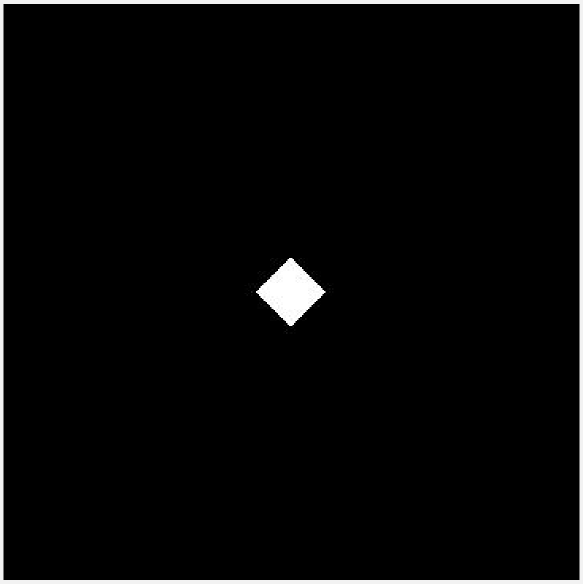
Μετάβαση στον κώδικα MATLAB

Αποτελέσματα & Συμπεράσματα

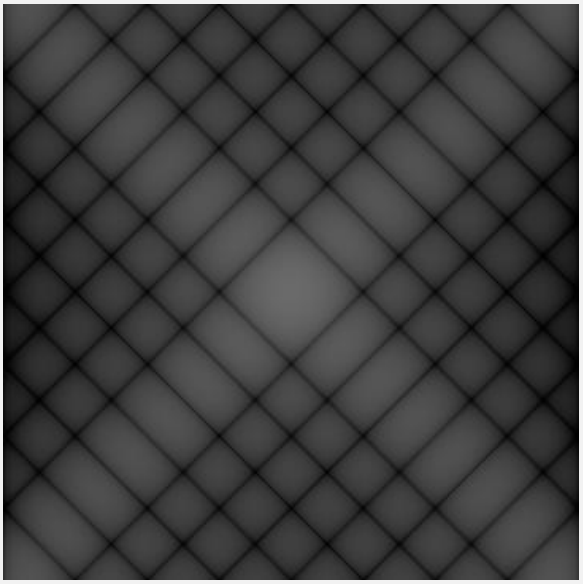
**ΑΡΧΙΚΗ ΕΙΚΟΝΑ (45°)
(10 x 10)**



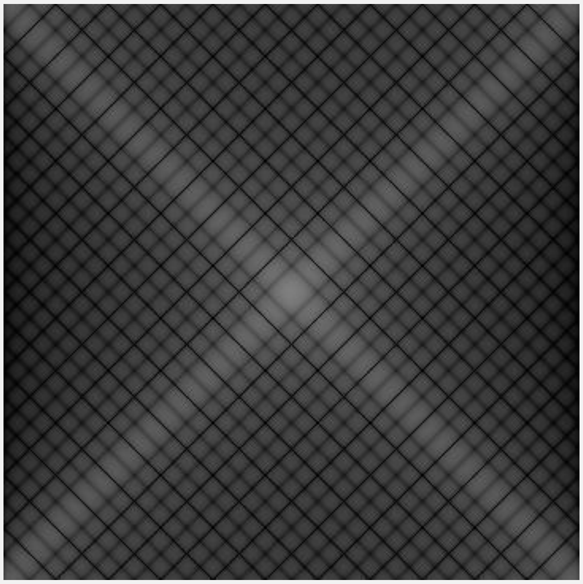
**ΑΡΧΙΚΗ ΕΙΚΟΝΑ (45°)
(30 x 30)**



**ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER
(10 x 10)**



**ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER
(30 x 30)**



Παρατηρούμε ότι κατά τη περιστροφή της αρχικής μας εικόνας, περιστρέφεται και ο μετασχηματισμός Fourier κατά την ίδια γωνία. Απτη θεωρία αυτό επαληθεύεται από το θεώρημα της περιστροφής:

$$f(r, \theta - \theta_o) \Leftrightarrow F(\omega, \theta + \theta_o)$$

5 – M.F ΜΕΤΑΤ. ΤΕΧΝΙΤΗΣ ΕΙΚΟΝΑΣ

Και στη 5^{η} άσκηση μας ζητείται να επαναλάβουμε τη διαδικασία με τη διαφορά ότι αυτή τη φορά μετατοπίζουμε το κεντρικό παράθυρο ώστε να μη βρίσκεται πλέον στο κέντρο της εικόνας.

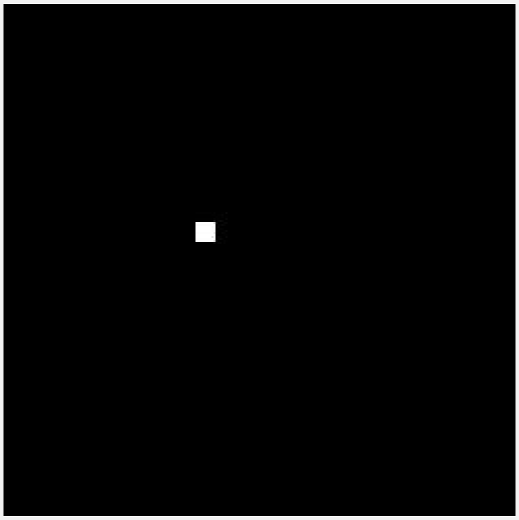
Θεωρητική Ανάλυση – Υλοποίηση

Η υλοποίηση είναι ακριβώς η ίδια με προηγουμένως. Η μόνη διαφορά είναι ότι χρησιμοποιούμε τιμές offset (στον οριζόντιο και κάθετο άξονα) που παράγονται από γεννήτρια τυχαίων αριθμών για να μετακινηθεί το κεντρικό παράθυρο προς μια κατεύθυνση.

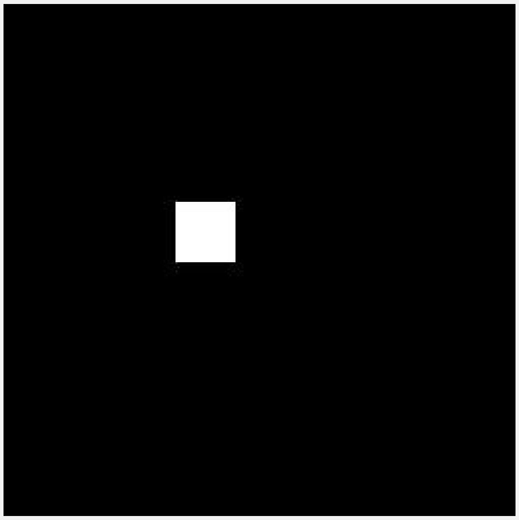
Μετάβαση στον κώδικα MATLAB

Αποτελέσματα & Συμπεράσματα

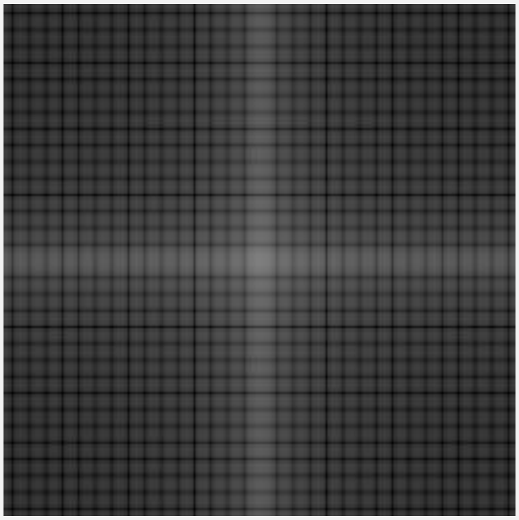
ΑΡΧΙΚΗ ΕΙΚΟΝΑ (ΜΕΤΑΤΟΠ.) (10 x 10)



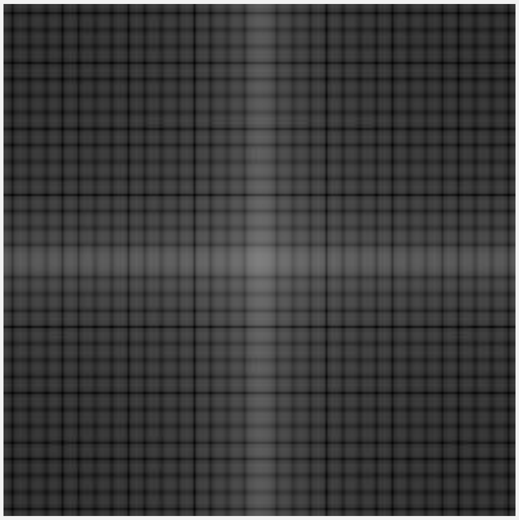
ΑΡΧΙΚΗ ΕΙΚΟΝΑ (ΜΕΤΑΤΟΠ.) (30 x 30)



ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER (10 x 10)



ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER (30 x 30)



Παρατηρούμε ότι ο μετασχηματισμός των δύο εικόνων με μετατοπισμένο κεντρικό τετράγωνο είναι ίδιος με αυτόν στη 1η περίπτωση. Καταλήγουμε δηλαδή στο συμπέρασμα ότι ο μετασχηματισμός Fourier δεν επηρεάζεται από πιθανές μετατοπίσεις, είναι δηλαδή shift-invariant, γεγονός που επιβεβαιώνεται από τη θεωρία.

$$f(x,y)e^{-j2\pi(ux/M+vy/N)} \Leftrightarrow F(u-u_o, v-v_o)$$

και:

$$F(x-x_o, y-y_o) \Leftrightarrow f(u,v)e^{-j2\pi(ux_o/M+vy_o/N)}$$

6 – M.F ΙΣΟΣΤΑΘΜΙΣΜ. ΕΙΚΟΝΑΣ

Στην 6η και τελευταία άσκηση μας ρωτήθηκε εάν η ισοστάθμιση της εικόνας θα αλλάξει το φάσμα της.

Θεωρητική Ανάλυση – Υλοποίηση

Διαβάζουμε όπως το πρώτο ερώτημα την εικόνα. Την ισοσταθμίζουμε και εφαρμόζουμε μετασχηματισμό Fourier τόσο στην αρχική όσο και στην ισοσταθμισμένη. Τέλος, συγκρίνουμε τα αποτελέσματά τους.

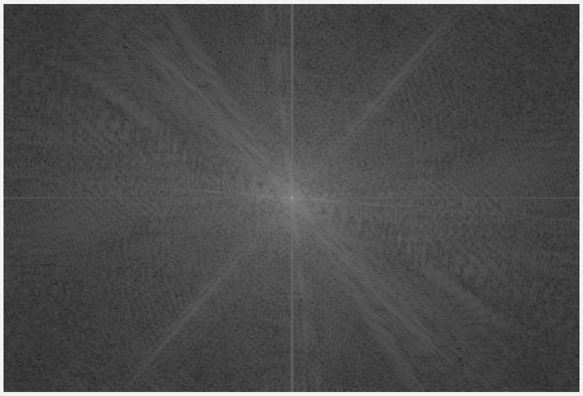
Μετάβαση στον κώδικα MATLAB

Αποτελέσματα & Συμπεράσματα

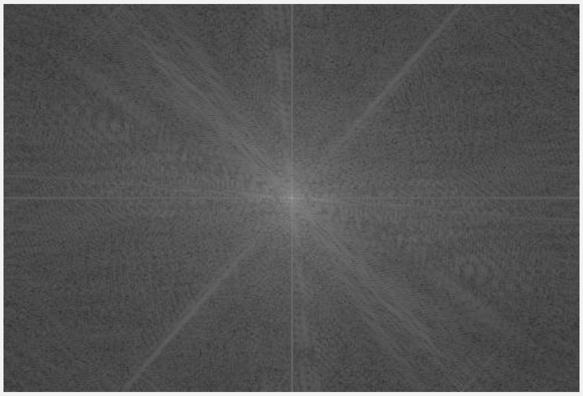
ΑΡΧΙΚΗ ΕΙΚΟΝΑ

ΙΣΟΣΤΑΘΜΙΣΜΕΝΗ ΕΙΚΟΝΑ

ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ ΑΡΧΙΚΗΣ ΕΙΚΟΝΑΣ



ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ ΙΣΟΣΤΑΘΜΙΣΜΕΝΗΣ ΕΙΚΟΝΑΣ



Παρατηρούμε με μια πρώτη ματιά, ότι ο μετασχηματισμός Fourier μεταβάλλεται ελάχιστα. Ωστόσο, γνωρίζοντας τη θεωρία, μπορούμε να παρατηρήσουμε ότι μετά την ισοστάθμισης της εικόνας, το φάσμα καταλαμβάνει μεγαλύτερη έκταση.

ΠΑΡΑΡΤΗΜΑ – ΚΩΔΙΚΑΣ

Σε περίπτωση που χρειαστεί ο κώδικας τον παραθέτουμε παρακάτω:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% BASIC INFORMATION                                   %
% Course: Digital Image Processing - Lab 3            %
% Deadline: 09-04-2019                                %
% LAB31239720:  Pantelis Karamailis, 2016030040       %
%               Kostantinos Vlachos, 2016030042       %
%               Mixalis Galanis,     2016030036       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Clearing things up
close all;
clearvars;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Exercise 1 - FOURIER TRANSFORMATION OF TOOLS IMAGE   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
original_image_1_1 = imread('tools.bmp'); %Gathering Image (Resolution:
510x343)
%Applying Fourier Transformation to the image without shifting
F_1_1 = abs(fft2(original_image_1_1));
%Applying Fourier Transformation to the image with shifting
F_1_2 = abs(fftshift(fft2(original_image_1_1)));
%Applying Log Filters
c_1_2 = 0.04; % 0.02 < c < 0.07 works best for most spectrum detail in
grayscale
Fourier_image_1_1 = c_1_2 * log(1 + F_1_1);
Fourier_image_1_2 = c_1_2 * log(1 + F_1_2);
%Using Color Map to Display Fourier Transformation
colormap(jet());
c_1_3 = 4; % 3 < c < 4.5 works best for most spectrum detail with color map
Fourier_image_1_3 = c_1_3 * log(1 + F_1_2);
%Displaying and comparing images
figure(1);
subplot(2,2,1); imshow(original_image_1_1); title("ORIGINAL IMAGE");
subplot(2,2,2); imshow(F_1_1); title("FOURIER TRANSFORM OF IMAGE (NO
SHIFT)");
subplot(2,2,3); imshow(Fourier_image_1_2); title("FOURIER TRANSFORM OF IMAGE
(SHIFT, GRAY)");
subplot(2,2,4); image(Fourier_image_1_3); title("FOURIER TRANSFORM OF IMAGE
(SHIFT, RGB)");
%imwrite(Fourier_image_1_1, 'Fourier_image_1_1.jpg');
%imwrite(Fourier_image_1_2, 'Fourier_image_1_2.jpg');
```

Μετάβαση στα αποτελέσματα της άσκησης 1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Exercise 2 - FOURIER TRANSFORMATION OF IMAGE PATTERS %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
original_image_2_1 = (imread('horizontal_lines_pattern.jpg'));
original_image_2_2 = imread('vertical_lines_pattern.jpg');
%Applying Fourier Transformation to the image with shifting
F_2_1 = abs(fftshift(fft2(original_image_2_1)));
F_2_2 = abs(fftshift(fft2(original_image_2_2)));
%Applying Log Filters
c_2_1 = 0.04; % 0.02 < c < 0.07 works best for most spectrum detail in
grayscale
c_2_2 = 0.02; % 0.02 < c < 0.07 works best for most spectrum detail in
grayscale
Fourier_image_2_1 = c_2_1 * log(1 + F_2_1);
Fourier_image_2_2 = c_2_2 * log(1 + F_2_2);
%Displaying and comparing images
figure(2);
subplot(1,2,1); imshow(original_image_2_1); title("HORIZONTAL LINES
(ORIGINAL)");
subplot(1,2,2); imshow(Fourier_image_2_1); title("HORIZONTAL LINES (FOURIER
TRANSFORM)");
figure(3);
subplot(1,2,1); imshow(original_image_2_2); title("VERTICAL LINES
(ORIGINAL)");
subplot(1,2,2); imshow(Fourier_image_2_2); title("VERTICAL LINES (FOURIER
TRANSFORM)");
%imwrite(Fourier_image_2_1, 'Fourier_image_2_1.jpg');
%imwrite(Fourier_image_2_2, 'Fourier_image_2_2.jpg');
```

Μετάβαση στα αποτελέσματα της άσκησης 2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Exercise 3 - FOURIER TRANSFORMATION FOR BLACK IMAGE  %
%              WITH WHITE BOX                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Creating image size
image_resolution = 256;
diameters = [10 30];
%Analyzing Pixel Data
for m = 1 : length(diameters) %For every diameter
    %Creating image and target area
    image = zeros(image_resolution);
    target_space_start = (image_resolution - diameters(m))/2;
    target_space_end = (image_resolution + diameters(m))/2;
    %Filling White Box
    for i = 1 : length(image) %For every pixel in horizontal axis
        for j = 1 : length(image) %For every pixel in vertical axis
            if (i >= target_space_start && i <= target_space_end && j >=
target_space_start && j <= target_space_end)
                image(i,j) = 255;
            end
        end
```

```
    end
    %Applying Fourier Transformation & Log Filters
    F_3_1 = abs(fftshift(fft2(image)));
    c_3_1 = 0.04; % 0.02 < c < 0.07 works best for most spectrum detail in
grayscale
    Fourier_image_3_1 =  c_3_1 * log(1 + F_3_1);
    %Displaying and comparing images
    figure(4);
    subplot(2,2,2*m - 1); imshow(image); title([num2str(diameters(m)) 'x'
num2str(diameters(m)) ' diameter box (ORIGINAL)']);
    subplot(2,2,2*m); imshow(Fourier_image_3_1); title([num2str(diameters(m))
'x' num2str(diameters(m)) ' diameter box (FOURIER TRANSFORM)']);
    %imwrite(image, ['Fourier_image_3_1_' num2str(2*m - 1) '.jpg']);
    %imwrite(Fourier_image_3_1, ['Fourier_image_3_1_' num2str(2*m) '.jpg']);
end
```

Μετάβαση στα αποτελέσματα της άσκησης 3

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Exercise 4 - FOURIER TRANSFORMATION FOR BLACK IMAGE  %
%              WITH ROTATED WHITE BOX                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Creating image size
image_resolution = 256;
diameters = [10 30];
%Analyzing Pixel Data
for m = 1 : length(diameters) %For every diameter
    %Creating image and target area
    image = zeros(image_resolution);
    target_space_start = (image_resolution - diameters(m))/2;
    target_space_end = (image_resolution + diameters(m))/2;
    %Filling White Box
    for i = 1 : length(image) %For every pixel in horizontal axis
        for j = 1 : length(image) %For every pixel in vertical axis
            if (i >= target_space_start && i <= target_space_end && j >=
target_space_start && j <= target_space_end)
                image(i,j) = 255;
            end
        end
    end
    %Rotating image first
    image = imrotate(image, 45);
    %Applying Fourier Transformation & Log Filters
    F_4_1 = abs(fftshift(fft2(image)));
    c_4_1 = 0.04; % 0.02 < c < 0.07 works best for most spectrum detail in
grayscale
    Fourier_image_4_1 =  c_4_1 * log(1 + F_4_1);
    %Displaying and comparing images
    figure(5);
    subplot(2,2,2*m - 1); imshow(image); title([num2str(diameters(m)) 'x'
num2str(diameters(m)) ' diameter box rotated (ORIGINAL)']);
```

```
    subplot(2,2,2*m); imshow(Fourier_image_4_1); title([num2str(diameters(m))
'x' num2str(diameters(m)) ' diameter box rotated (FOURIER TRANSFORM)']);
    %imwrite(image, ['Fourier_image_4_1_' num2str(2*m - 1) '.jpg']);
    %imwrite(Fourier_image_4_1, ['Fourier_image_4_1_' num2str(2*m) '.jpg']);
end
```

Μετάβαση στα αποτελέσματα της άσκησης 4

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Exercise 5 - FOURIER TRANSFORMATION FOR BLACK IMAGE  %
%               WITH RELOCATED WHITE BOX               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Creating image size
image_resolution = 256;
diameters = [10 30];
%Generating random offsets
random_offsets = zeros(1,2);
for k = 1 : 2 % 1 random offset for X axis, 1 random offset for Y axis
    random_offsets(k) = (rand - 0.5) * (image_resolution - max(diameters));
end
%Analyzing Pixel Data
for m = 1 : length(diameters) %For every diameter
    %Creating image and target area with random offset
    image = zeros(image_resolution);
    target_space_start_X = (image_resolution - diameters(m))/2 +
random_offsets(1);
    target_space_end_X = (image_resolution + diameters(m))/2 +
random_offsets(1);
    target_space_start_Y = (image_resolution - diameters(m))/2 +
random_offsets(2);
    target_space_end_Y = (image_resolution + diameters(m))/2 +
random_offsets(2);
    %Filling White Box
    for i = 1 : length(image) %For every pixel in horizontal axis
        for j = 1 : length(image) %For every pixel in vertical axis
            if (i >= target_space_start_X && i <= target_space_end_X && j >=
target_space_start_Y && j <= target_space_end_Y)
                image(i,j) = 255;
            end
        end
    end
    %Applying Fourier Transformation & Log Filters
    F_5_1 = abs(fftshift(fft2(image)));
    c_5_1 = 0.04; % 0.02 < c < 0.07 works best for most spectrum detail in
grayscale
    Fourier_image_5_1 =  c_5_1 * log(1 + F_3_1);
    %Displaying and comparing images
    figure(6);
    subplot(2,2,2*m - 1); imshow(image); title([num2str(diameters(m)) 'x'
num2str(diameters(m)) ' diameter box rotated (ORIGINAL)']);
    subplot(2,2,2*m); imshow(Fourier_image_5_1); title([num2str(diameters(m))
'x' num2str(diameters(m)) ' diameter box rotated (FOURIER TRANSFORM)']);
```

```
    %imwrite(image, ['Fourier_image_5_1_' num2str(2*m - 1) '.jpg']);
    %imwrite(Fourier_image_5_1, ['Fourier_image_5_1_' num2str(2*m) '.jpg']);
end
```

Μετάβαση στα αποτελέσματα της άσκησης 5

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Exercise 6 - DIFFERENCE BETWEEN ORIGINAL & EQUALIZED %
%               IMAGE                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
original_image_6_1 = imread('tools.bmp'); %Gathering Image (Resolution:
510x343)
equalized_image_6_2 = histeq(original_image_6_1);
%Applying Fourier Transformations to the images with shifting
F_6_1 = abs(fftshift(fft2(original_image_6_1)));
F_6_2 = abs(fftshift(fft2(equalized_image_6_2)));
%Applying Log Filters
c_6_1 = 0.04; % 0.02 < c < 0.07 works best for most spectrum detail in
grayscale
c_6_2 = 0.04; % 0.02 < c < 0.07 works best for most spectrum detail in
grayscale
Fourier_image_6_1 = c_6_1 * log(1 + F_6_1);
Fourier_image_6_2 = c_6_2 * log(1 + F_6_2);
%Displaying and comparing images
figure(7);
subplot(2,2,1); imshow(original_image_6_1); title("ORIGINAL IMAGE");
subplot(2,2,2); imshow(equalized_image_6_2); title("EQUALIZED IMAGE");
subplot(2,2,3); imshow(Fourier_image_6_1); title("FOURIER TRANSFORM OF
ORIGINAL IMAGE");
subplot(2,2,4); imshow(Fourier_image_6_2); title("FOURIER TRANSFORM OF
EQUALIZED IMAGE");
%imwrite(equalized_image_6_2, ['equalized_image_6_1.jpg']);
%imwrite(Fourier_image_6_1, ['Fourier_image_6_2.jpg']);
%imwrite(Fourier_image_6_2, ['Fourier_image_6_3.jpg']);
```

Μετάβαση στα αποτελέσματα της άσκησης 6