Επεξεργασία Εικόνας

**Εργασία 2η**



**Η εργασία θα πραγματοποιηθεί από τον φοιτητή:**

**Όνομα:** Κωνσταντίνος

**Επώνυμο:** Σταθακόπουλος

**ΑΜ:** 161041

**Τμήμα:** IP Lab Group 1 (Τρίτη 11:00-13:00)

# Ερωτήμα 1ο

Ακολουθεί ο κώδικας ανίχνευσης ακμών με τις ζητούμενες μεθόδους και στην συνέχεια ακολουθούν οι έξοδοι που δίνει.

**Κώδικας :**

im=imread('cameraman.tif'); %reading the image

C=imnoise(im,'gaussian',0,0.05); % image with noise

edge\_p=edge(C,'prewitt'); %i)

subplot(2,3,1)

imshow(edge\_p)

title('prewitt')

edge\_r=edge(C,'roberts'); %ii)

subplot(2,3,2)

imshow(edge\_r)

title('roberts')

edge\_s=edge(C,'sobel'); %iii)

subplot(2,3,3)

imshow(edge\_s)

title('sobel')

l=fspecial('laplacian',0); %iv)

ic\_l=filter2(l,C);

subplot(2,3,4)

imshow(mat2gray(ic\_l))

title('Z. crossings(Laplacian)')

fspecial('log',13,2) %v)

subplot(2,3,5)

g1=edge(C,'log');

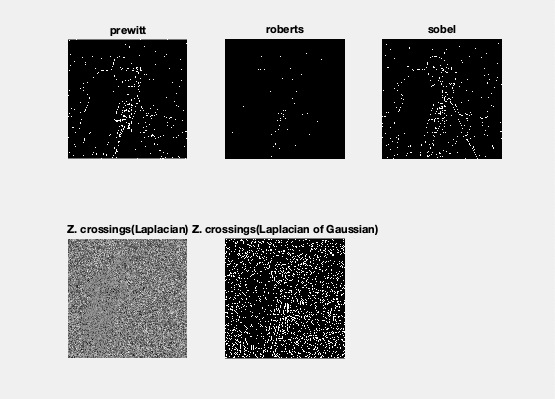
imshow(g1)

title('Z. crossings(Laplacian of Gaussian)')

**Παρατηρήσεις**

Όπως φαίνεται στο παρακάτω στιγμιότυπο οθόνης **η καλύτερη μέθοδος ανίχνευσης ακμών** είναι η sobel γιατί φαίνεται καλύτερα η φιγούρα του φωτογράφου. **Η χειρότερη μέθοδος ανίχνευσης ακμών** είναι η zero crossings laplacian οπου μετέτρεψε την εικόνα σε θόρυβο χωρίς να φαίνεται τι απεικονίζει. Η μέθοδος prewitt είναι κοντά στην καλύτερη μέθοδο αλλά δεν είναι τόσο ακριβής και οι Roberts και zero crossings laplacian of Gaussian είναι λίγο καλύτερες από την χειρότερη μέθοδο δίοτι πάλι δεν έιναι ξεκάθαρο το τι απεικονίζει η εικόνα.

**Έξοδοι :**



**Βασική αρχή της μεθόδου canny:**

1. Γίνεται ανίχνευση άκρου με χαμηλό ρυθμό σφάλματος, πράγμα που σημαίνει ότι η ανίχνευση πρέπει να αγγίζει με ακρίβεια όσες άκρες εμφανίζονται στην εικόνα όσο το δυνατόν
2. Το σημείο ακμής που ανιχνεύεται από τον χειριστή θα πρέπει να εντοπίζεται με ακρίβεια στο κέντρο της άκρης.
3. Μια δεδομένη άκρη της εικόνας πρέπει να επισημαίνεται μόνο μία φορά και, όπου είναι δυνατόν, ο θόρυβος της εικόνας δεν πρέπει να δημιουργεί ψευδείς άκρες.

**Κώδικας :**

%canny method

figure

c1=edge(C,'canny',0.2,0.2);

subplot(2,3,1)

imshow(c1)

title('T=0.2 & S=0.2')

c2=edge(C,'canny',0.2,2);

subplot(2,3,2)

imshow(c2)

title('T=0.2 & S=2')

c3=edge(C,'canny',0.3,2.5);

subplot(2,3,3)

imshow(c3)

title('T=0.3 & S=2.5')

c4=edge(C,'canny',0.5,3);

subplot(2,3,4)

imshow(c4)

title('T=0.5 & S=3')

c5=edge(C,'canny',0.1,5);

subplot(2,3,5)

imshow(c5)

title('T=0.1 & S=5')

c6=edge(C,'canny',0.9,1);

subplot(2,3,6)

imshow(c6)

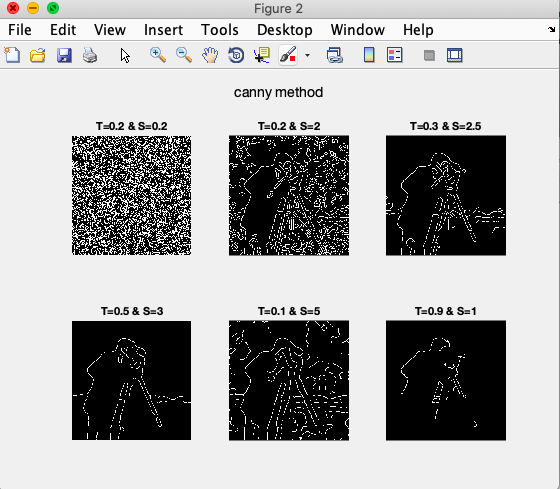
title('T=0.9 & S=1')

suptitle('canny method')

**Αποτελέσματα:**

Όπως φαίνεται και παρακάτω αν οι τιμές Threshold και sigma είναι πολύ χαμηλές έχουμε μια εικόνα που είναι γεμάτη θόρυβο, αν όμως έχουμε μικρή τιμή Threshold και λίγο μεγαλύτερη τιμή sigma (όπως στην 2η περίπτωση) τότε έχουμε καλύτερο αποτέλεσμα και αν αυξηθούν λίγο αυτές οι τιμές με την ίδια αναλόγια έχουμε την καλύτερη ανίχνευση ακμών από όλες τις άλλες δοκιμές. Τέλος παρατηρούμε ότι όσο μεγαλώνει η τιμή Threshold και μικρένει η τιμή sigma (και το ανάποδο) βλέπουμε ότι πάλι το αποτέλεσμα δεν είναι αυτό που θέλουμε.

**Έξοδοι :**



# Ερωτήμα 2ο

Ακολουθεί ο κώδικας για την συνάρτηση myEdge, μετά οι δοκιμές της συνάρτησης και στην συνέχεια ακολουθούν οι έξοδοι που δίνει.

**Κώδικας :**

function[x] = myEdge(x,filter)

if strcmp(filter,'roberts') == 1 %creating table X1.

x1 = [1 0 0; 0 -1 0; 0 0 0];

%creatign filter by using X1 table

xx1 = filter2(x1,x); %creating table X2.

x2 = [0 1 0; -1 0 0; 0 0 0];

%creatign filter by using X2 table

xx2 = filter2(x2,x);

%combination of filters and elevation in square.

rob = sqrt(xx1.^2 + xx2.^2);

[x]=rob/255;

end

if strcmp(filter,'sobel') == 1 %creating table X1.

x1 = [1 0 0; 0 -1 0; 0 0 0];

%creatign filter by using X1 table

xx1 = filter2(x1,x); %creating table X2.

x2 = x1';

%creatign filter by using X2 table

xx2 = filter2(x2,x);

%combination of filters and elevation in square.

sob = sqrt(xx1.^2 + xx2.^2);

[x]=sob/255;

end

c=imread('cameraman.tif');

t=imread('tire.tif');

c1=myEdge(c,'roberts');

subplot(2,2,1)

imshow(c1)

title('Custom roberts filter')

c2=edge(c,'roberts');

subplot(2,2,2)

imshow(c2)

title('Original roberts filter')

t1=myEdge(t,'roberts');

subplot(2,2,3)

imshow(t1)

title('Custom roberts filter')

t2=edge(t,'roberts');

subplot(2,2,4)

imshow(t2)

title('Original roberts filter'),figure

c1=myEdge(c,'sobel');

subplot(2,2,1)

imshow(c1)

title('Custom sobel filter')

c2=edge(c,'sobel');

subplot(2,2,2)

imshow(c2)

title('Original sobel filter')

t1=myEdge(t,'sobel');

subplot(2,2,3)

imshow(t1)

title('Custom sobel filter')

t2=edge(t,'sobel');

subplot(2,2,4)

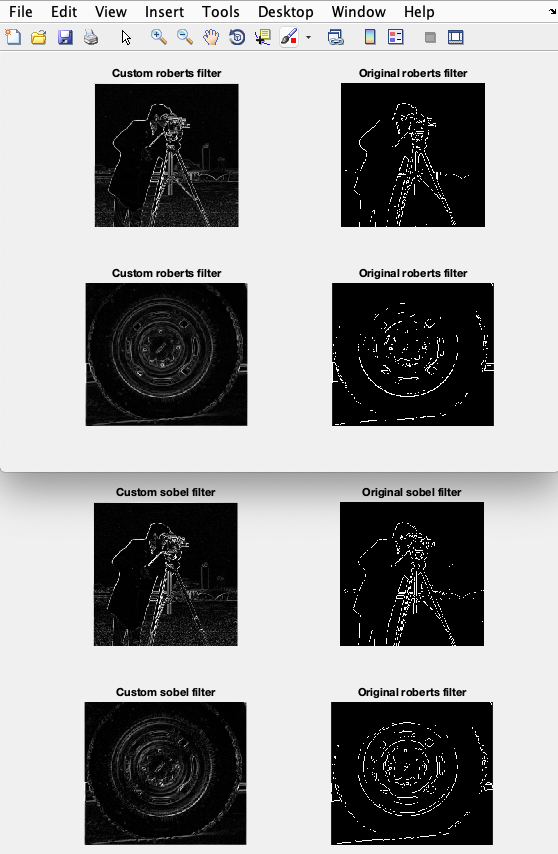
imshow(t2)

title('Original roberts filter')

**Αποτελέσματα:**

Τα αποτελέσματα διαφέρουν αρκετά μεταξύ τους. Η μέθοδος Robert που φτιάξαμε, εμφανίζει την εικόνα πολύ πιο καθαρή και με περισσότερη ακρίβεια από ότι η μέθοδος με χρήση της edge. Επισής, η μέθοδος Sobel που φτιάξαμε, εμφανίζει την εικόνα πολύ πιο καθαρή, με περισσότερη ακρίβεια και με πιο έντονα τονισμένα τα περιγράμματα που έχει από ότι η αρχική.

**Έξοδοι :**



# Ερωτήμα 3ο

Ακολουθεί ο κώδικας για την δοκιμή των ζητούμενων φίλτρων με το μετασχηματισμό Fourier (οι συνάρτησεις fftshow, lbutter και hbutter έχουν υλοποιηθεί σε ξεχωριστά αρχεία) μετά οι έξοδοι που δίνει και τέλος οι παρατηρήσεις.

**Κώδικας :**

im=imread('cameraman.tif');

cf=fftshift(fft2(im)); %rearranges a Fourier transform

%Low pass filter

[x,y]=meshgrid(-128:217,-128:127); %we create the circle first

z=sqrt(x.^2+y.^2);

c=(z<50); %circle matrix

cfl=cf\*c; %with the operator .\* matlab gives error for the dimensions

subplot(2,2,1)

fftshow(cfl,'log') %function to show the spectrum

title('Low pass filter');

cfli=ifft2(cfl); %Applying ideal low pass filtering and inverting image

subplot(2,2,2)

fftshow(cfli,'abs')

title('Low pass filter after inversion');

%High pass filter

[x,y]=meshgrid(-128:127,-128:127); %we create the circle first

z=sqrt(x.^2+y.^2);

c=(z>50); %circle matrix

cfh=cf.\*c; %multiplying the circle matrix by the DFT of the image

subplot(2,2,3)

fftshow(cfh,'log')

cfhi=ifft2(cfh);

title('High pass filter');

subplot(2,2,4)

fftshow(cfhi,'abs')

title('High pass filter after inversion');

%Low pass butterworth filter

[x,y]=meshgrid(-128:217,-128:127);

bl=1./(1+((x.^2+y.^2)/15).^2);

bl=lbutter(c,15,1);

cfbl=cf.\*bl;

figure,subplot(2,2,1)

fftshow(cfbl,'log')

title('Low pass butterworth filter')

cfbli=ifft2(cfbl);

subplot(2,2,2)

fftshow(cfbli,'abs') %inverted to get the end result

title('Low pass butterworth filter after inversion');

%High pass butterworth filter

bh=hbutter(im,15,1);

cfbh=cf.\*bh;

subplot(2,2,3)

fftshow(cfbh,'log')

title('High pass butterworth filter')

cfbhi=ifft2(cfbh); %inverted to get the end result

subplot(2,2,4)

fftshow(cfbhi,'abs')

title('High pass butterworth filter after inversion')

%Low pass Gaussian filter

g1=mat2gray(fspecial('gaussian',256,10));

cg1=cf.\*g1;

figure,subplot(2,2,1)

fftshow(cg1,'log')

title('Low pass Gaussian filter')

cgi1=ifft2(cg1);

subplot(2,2,2)

fftshow(cgi1,'abs');

title('Low pass Gaussian filter after inversion')

%High pass Gaussian filter

h1=1-g1;

ch1=cf.\*h1;

subplot(2,2,3)

fftshow(ch1,'log')

ch1i=ifft2(ch1);

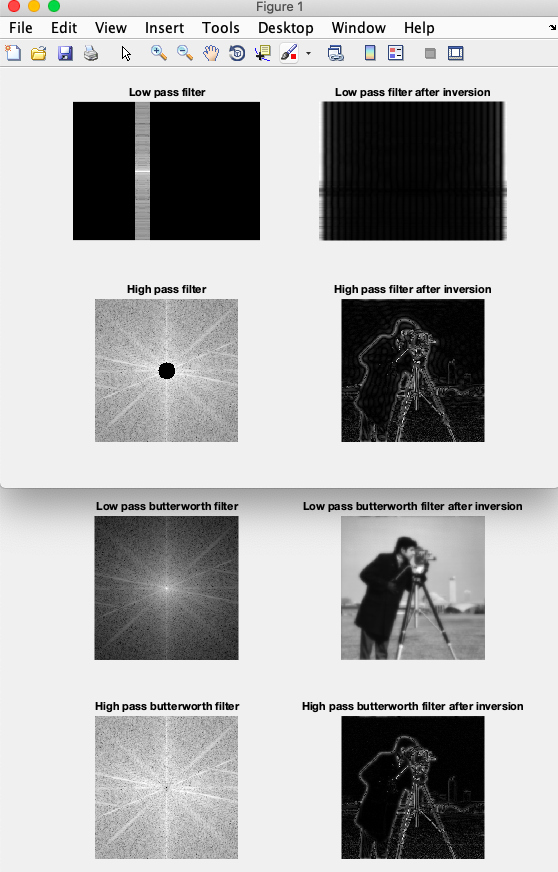
subplot(2,2,4)

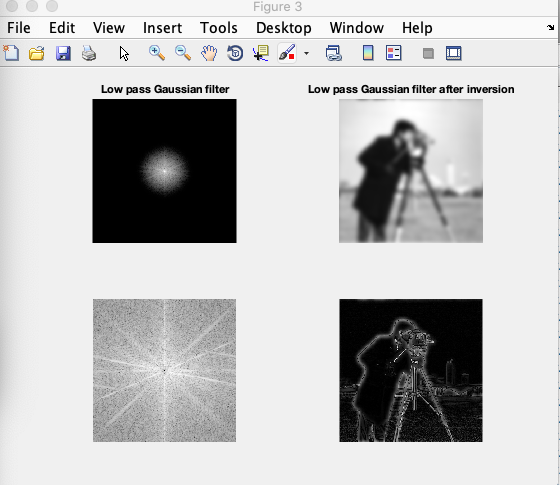
fftshow(ch1i,'abs')

**Παρατηρήσεις**

Στα ιδεατά φίλτρα είχα σαν “threshold” την τιμή 15 (εδώ το βαθυπέρατο φίλτρο είχε ένα λάθος που δεν κατάφερα να εντοπίσω). Το ίδιο έκανα και στα Butterworth φίλτρα. Αν αλλάξουμε την τιμή αυτή έχουμε διαφορετικό αποτέλεσμα ανάλαγα πόσο μεγάλη είναι η τιμή αυτή. Στα Butterworth φίλτρα είναι πιο εμφανής οι λεπτομέρειες της εικόνας, φαίνονται πιο καθαρά. Τα ιδεατά φίλτρα θολώνουν λίγο την εικόνα, ειδικά στον IFFT2 μετασχηματισμό Fourier. Στα φίλτρα Gaussian η εικόνα είναι λίγο πιο καθαρή σχέση με τα προηγούμενα δύο φίλτρα όπως φαίνεται παρακάτω.

**Έξοδοι :**

****

****

# Ερωτήμα 4ο

Ακολουθεί ο κώδικας του προγραμματος με τα ζητούμενα και στην συνέχεια ακολουθούν οι έξοδοι που δίνει.

**Κώδικας :**

im=double(imread('cameraman.tif')); %in this form in order to be able to use the log2 function

d=10;

n=2;

subplot(1,2,1)

imshow(im./255);

title('Initial image')

%Butterworth high pass filter

A=zeros(r,c);

for i=1:r

for j=1:c

A(i,j)=(((i-r/2).^2+(j-c/2).^2)).^(.5);

H(i,j)=1/(1+((d/A(i,j))^(2\*n)));

end

end

%Using it for my application as homomorphic filtering is

%application specific, taking the value of alphaL and alphaH

%values accordingly.

[r c]=size(im);

alphaL=.0999;

aplhaH=1.01;

H=((aplhaH-alphaL).\*H)+alphaL;

H=1-H;

%log of image

im\_l=log2(1+im);

%DFT of logged image

im\_f=fft2(im\_l);

%Filter Applying DFT image

im\_nf=H.\*im\_f;

%Inverse DFT of filtered image

im\_n=abs(ifft2(im\_nf));

%Inverse log

im\_e=exp(im\_n);

subplot(1,2,2)

imshow((im\_e),[]) %Displays the grayscale image I, scaling the display based on the range of pixel values in I

title('Image after using homomorphic filter')

**Έξοδοι :**

****

# Ερωτήμα 5ο

Ακολουθεί ο κώδικας του προγραμματος με τα ζητούμενα και στην συνέχεια ακολουθούν οι έξοδοι που δίνει.

**Κώδικας :**

im=imread('circbw.tif');

%dilation

sq=ones(3,3) %3x3 square

di = [1 0 0;1 1 1;0 0 1]; %diamond

sqh = [1 1 1;1 0 1;1 1 1]; %square with 0 in the middle

%the asked operations

dial = imdilate(im,sq);

ero = imerode(im,sq);

open = imopen(im,sq);

close = imclose(im,sq);

%showing the results

subplot(3,4,1)

imshow(dial); title({'Dilation','Square'}); subplot(3,4,2)

imshow(ero); title({'Erosion','Square'}); subplot(3,4,3)

imshow(open); title({'Open','Square'}); subplot(3,4,4)

imshow(close); title({'Close','Square'});

%the asked operations

dial = imdilate(im,di);

ero = imerode(im,di);

open = imopen(im,di);

close = imclose(im,di);

%showing the results

subplot(3,4,5)

imshow(dial); title({'Dilation','Diamond'}); subplot(3,4,6)

imshow(ero); title({'Erosion','Diamond'}); subplot(3,4,7)

imshow(open); title({'Open','Diamond'}); subplot(3,4,8)

imshow(close); title({'Close','Diamond'});

%the asked operations

dial = imdilate(im,sqh);

ero = imerode(im,sqh);

open = imopen(im,sqh);

close = imclose(im,sqh);

%showing the results

subplot(3,4,9)

imshow(dial);

title({'Dilation','S. 0 in the middle'}); subplot(3,4,10)

imshow(ero);

title({'Erosion','S. 0 in the middle'}); subplot(3,4,11)

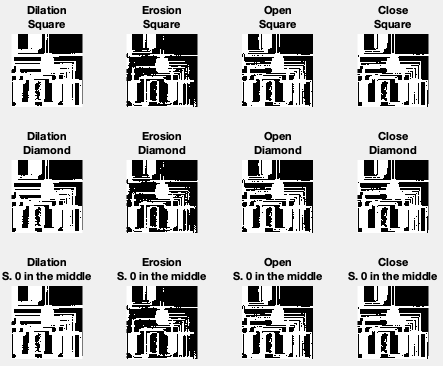
imshow(open);

title({'Open','S. 0 in the middle'}); subplot(3,4,12)

imshow(close);

title({'Close','S. 0 in the middle'});

**Έξοδοι :**

****

# Ερωτήμα 6ο

Ακολουθεί ο κώδικας του προγραμματος με τα ζητούμενα και στην συνέχεια ακολουθούν οι έξοδοι που δίνει.

**Κώδικας :**

im=imread('circles\_and\_lines.jpg');

%di = strel('disk',6); % with bigger number (>=7) the image becomes black

%open = imopen(im,di); %disk-shaped structuring element with a radius of 5 pixels

%this structure doesn't give the best result

line = strel('line',20,90); %20-90 gives the best result

open = imopen(im,line);

%imshow(open)

imwrite(open,'lines.png')

sp = strel('sphere',6); % with bigger number (>=7) the image becomes black

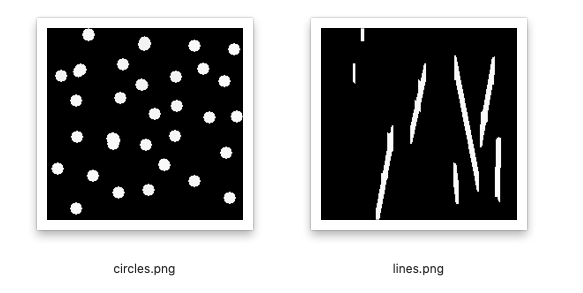
%six gives almost the same circles we have in the initial image

open = imopen(im,sp);

imwrite(open,'circles.png')

%imshow(open)

**Έξοδοι :**

****

**Παρατηρήσεις**

Μετά από αρκετό πειραματισμό με δίαφορα structuring elements καταλήγουμε ότι τα καλύτερα αποτελέσματα τα έχουμε από τα sphere και line. Τα υπόλοιπα που δοκιμάστηκαν σε συνδυασμό με δίαφορα μεγέθη δεν έδωσαν σωστά απότελεσματα (π.χ. το rectangle έδωσε μια μόνο από τις γραμμές σωστά όχι άλλες). Επιπλεόν ήταν συγκεκριμένα τα μεγέθη που έδωσαν καλύτερο αποτέλεσμα γιατί για παράδειγμα στην σφαίρα αν δίναμε τιμή 7 θα είχαμε σαν αποτέλεσμα μαυρη εικόνα και στην γραμμή αν δίναμε άλλο συνδυασμό από 20-90 θα είχαμε λίγοτερες γραμμές που δεν ήταν κοντά στις γραμμές της αρχικής εικόνας.

# Ερωτήμα 7ο

**Εικόνα Β:** Erosion μορφολογική λειτουργία με δομικό στοιχείο diamond.

**Εικόνα Γ:** Erosion μορφολογική λειτουργία με δομικό στοιχείο square.

**Εικόνα Δ:** Open μορφολογική λειτουργία με δομικό στοιχείο line.

**Εικόνα Ε:** Dilation μορφολογική λειτουργία με δομικό στοιχείο square.