به نام خدا

محمدرضا پوررضا

اطلاعات گزارش

تاریخ: ۹۹/۱۰/۱۵

واژگان کلیدی:
مقاله
Wavelet Denoising
کوانتیزیشن
هار ترنسفورم
هرم گوسی
هرم ویولت

چکیده

در این تمرین ما به طور کلی با wavelet و هرم های گوسی و هرم های ویولت آشنا خواهیم شد و میتوانیم با استفاده از آنها به حذف نویز و فشرده سازی خواهیم پرداخت. در این تمرین از توابع از پیش آماده شده برای اعمال ویولت ترنسفورم و برعکس آن استفاده شده است ولی برای ایجاد هرم ها توابع استفاده شده همگی پیاده سازی شده و از توابع آماده استفاده نشده است.

۱-مقدمه

این گزارش شامل نحوه انجام هر یک از تمارین با استفاده از زبان MATLAB می‌باشد به همراه خروجی مربوط به هر یک از بخش های تمرین که در قالب png دریافت شده اند زیرا png و bitmap دو فرمتی هستند که lossless هستند در نتیجه کیفیت تصاویر کمتر نخواهد شد و همچنین برای تمرین هایی که نیاز به جدول و یا تصویر داشتند جداول ، تصاویر و تحلیل مربوط به هریک از این جدول ها و تصاویر قرار داده شده اند. توضیحات تشریحی مربوط به نحوه انجام حل هر یک از سوالات در بخش بعدی قرار داده شده است و بعد از آن سراغ تحلیل خروجی و در انتها کد مربوط به هر تمرین قرار داده شده است.

۲-توضیحات فنی تمرین

۲-۱- تمرین ۱-۱-۶

در این تمرین از ما خواسته شده است که برای یک تصویر به اندازه N*N تعیین کنیم که تا چند لول میتوانیم هرم را تشکیل دهیم تا به یک پیسکل برسیم در صورتی که N=2*j باشد. همانطور که میدانیم در هر مرحله اندازه تصویر نصف میشود در نتیجه اگر لول اول را برابر خود تصویر اصلی در نظر بگیریم برای اینکه به یک پیسکل برسیم به j+1 مرحله نیاز داریم. تعداد مجموع پیکسل هایی که هرم را تشکیل میدهند در صورتی که لول اخر یک پیکسل باشد فقط برابر با مجموع یک دنباله هندسی میشود که مجموع این لول ها برابر است با 4/3N*2 در نتیجه فقط به اندازه 1/3N*2 باید پیکسل بیشتر نگه داریم. همانطور که مشخص است این مقدار ، مقدار زیادی نیست در نتیجه با توجه به مزایایی که استفاده از هرم ها برای ما دارند که برابر هست با Alternative representation که میتوان از هر یک از تصاویر در لول های مختلف به عنوان

M1378.prz@gmail.com

اسکیل های مختلف آن تصویر استفاده کنیم یا از دیگر کاربرد های آن detail manipulation یا image compression است که کاربرد زیادی در پردازش تصاویر دارد در نتیجه این مقدار اضافی که باید تحمل کنیم زیاد قابل توجه نیست.و در نهایت از ما خواسته شده است که این هرم را برای یک تصویر تست ایجاد کنیم که تصویر تست ما تصویر لنا است و فیلتر استفاده شده برای اینکار فیلتر گوسین ۳ در ۳ است که نتایج اعمال آن بر روی تصویر در قسمت بعدی توضیح داده خواهد شد.

۲-۲ تمرین ۲-۱-۶

در این تمرین از ما خواسته شده است که با استفاده از یک فیلترباکس فیلتر دو در دو هرم را ایجاد کنیم برای اینکار ابتدا یک تابع به نام filter ایجاد میکنیم که یک تصویر را به عنوان ورودی دریافت میکند سپس بعد از اعمال این فیلتر بر روی تصویر با استفاده از downsample کردن به روش pixel removal اندازه تصویر را نصف خواهیم کرد و با استفاده از همین دو تابع میتوانیم تا سه مرحله هرم را ایجاد کنیم نکته دیگری که باید به آن توجه کنیم این است که در سوال از ما خواسته شده است هرم residual را نیز ایجاد کنیم که برای این کار قبل از down sample کردن تصویر ابتدا تفاضل تصویر بعد از اعمال فلیتر را با تصویر اولیه محاسبه میکنیم و با اینکار اولین سطح هرم residual بدست خواهد و به همین شکل برای تمامی سطوح این هرم مقادیر را بدست خواهیم آورد. سپس در بخش دوم سوال از ما خواسته شده است که با داشتن تصویر در بالاترین سطح هرم و با استفاده از upsample کردن به روش pixel replication سعی کنیم تا تصویر اصلی را ایجاد کنیم که برای این کار نیز یک تابع ایجاد شده است که تصویر را به عنوان ورودی دریافت میکند و سپس مقدار پیکسل موجود در سطر i و ستون j را در ۳ پیکسل مجاور خود قرار خواهد داد. در قسمت به تحلیل خروجی مربوط به این بخش خواهیم پرداخت و نتیجه را بررسی خواهیم کرد.

۲-۳ تمرین ۳-۱-۶

در این تمرین از ما خواسته شده است که همان تعداد سطوح هرم ینی ۳ سطح را که در مسئله قبلی ایجاد کرده بودیم را با استفاده از تبدیل Wavelet ایجاد کنیم و mother wavelet که در این سوال باید استفاده کنیم haar میباشد برای اینکار از تابع dwt2 با پارامتر هایی که یکی از آنها تصویر ورودی و دیگری haar است که مشخص کننده mother wavlet است استفاده خواهیم کرد خروجی این تابع ۴ تصویر با اندازه نصف تصویر اصلی خواهد بود شامل LL و LH ، HL و HH خواهد بود که تصویر LL تصویر اصلی با اندازه نصف تصویر قبلی است که برای ساخت هرم استفاده خواهد شد و سه تصویر دیگر لبه های تصویر در ۳ جهت افقی عمودی و قطری را خواهد داد و در مرحله بعد برای ساخت سطح بعدی هرم تصویر LL را به عنوان ورودی به تابع dwt2 خواهیم داد و همین کار را یک بار دیگر تکرار خواهیم کرد. و در نهایت از ما خواسته شده است که این دو روش را با یکدیگر مقایسه کنیم یک تفاوتی که بنظر میرسد این است که استفاده از باکس فیلتر دو در دو ممکن است نتواند فرکانس ماکسیسم تصویر را کاهش دهد و به خاطر همین دلیل ممکن است با پدیده Aliasing در استفاده از فیلتر باکس فیلتر دو در دو مواجه شویم در حالی که این پدیده در استفاده از mother wavelet هار رخ نخواهد داد.

۲-۴ تمرین ۴-۱-۶

در این تمرین از ما خواسته شده است که بر روی تمامی ضرایب هر ۴ تصویر خروجی از تابع dwt2 با مادر ویولت هار یک کوانتیزشن با فرمول داده شده را انجام دهیم برای اینکار در ابتدا بعد از خواندن تصویر و دادن آن به تابع dwt2 نتایج به دست آمده را به یک تابع خواهیم داد که در این تابع با استفاده فرمول داده شده در صورت سوال عمل کوانتیزشن را انجام خواهد برای متوجه شدن عملکرد این تابع میتوانیم برای خود مثالی بزنیم فرض کنیم مقدار یکی از ضرایب برابر با ۱۳ باشد در مرحله اول ابتدا این مقدار را بر دو تقسیم خواهیم کرد که برابر با ۶.۵ خواهد شد سپس این مقدار را به پایین گرد خواهیم کرد

درنیتجه مقدار ۶ را به برخواهد گرداند سپس این مقدار را در علامت این ضریب که مثبت است ضرب خواهیم کرد که تغییری در مقدار ۶ ایجاد نخواهد کرد سپس این مقدار را در دو ضرب خواهیم کرد که برابر با ۱۲ میشود همانطور که مشخص شد با استفاده از این تابع عمل کوانتیزشن را انجام خواهیم داد سپس این عمل را بر روی تمامی ضرایب هر ۴ تصویر اعمال خواهیم کرد و در نهایت ۴ تصویر خروجی را با استفاده از تابع `idwt2` دوباره `reconstruct` خواهیم کرد و در بخش بعدی خروجی این بخش را با تصویر اصلی مقایسه خواهیم کرد.

۵-۲ تمرین ۲-۶

در این تمرین از ما خواسته شده است که حداقل دو روش را برای از بین نویز در تصاویر نویزی با استفاده از wavelet معرفی کنیم. به طور کلی عمل denoising در تصاویر با استفاده از Wavelet به این شکل انجام میشود که در ابتدا با استفاده از از تابع dwt2 تبدیل ویولت را اعمال خواهیم کرد و سپس با استفاده از treshhold مقادیر کوچک را در تصاویر LH ، HL و HH حذف خواهیم کرد و تنها فرقی که روش های مختلف denoising با یکدیگر دارند تنها در اعمال treshhold میباشد حال دو روشی که ما برای حل این مسئله در نظر گرفته ایم دو روش معروف hard treshholding و روش soft treshholding است که در روش اول مقادیر کمتر از Treshhlod را صفر خواهیم کرد و مقادیر بزرگ تر از آن را هیچ تغییری نخواهیم داد و در روش دوم مانند روش اول مقادیر کمتر از treshhlod را صفر خواهیم کرد ولی تفاوت ان با روش قبلی در ضرایب بزرگتر از treshhold است که در این روش از تمامی این مقادیر به اندازه Treshhold کم خواهیم کرد. روش معمول در حذف نویز روش Soft treshholding است. حال برای تست این دو روش از ۳ تصویر استفاده شده است که یکی از آنها به نویز فلفل نمک و دیگری با نویز گوسین و تصویر آخری از یک شی در نور خیلی کم گرفته شده است و نتیجه اعمال denoising را بر روی این سه تصویر در بخش بعدی باهم بررسی خواهیم کرد.

۳- تحلیل خروجی ها

۱-۳ تمرین ۱-۱-۶

در این تمرین از ما خواسته شده بود که ساخت هرم گوسین را تا زمانی ادامه دهیم که به یک پیکسل بریم در این بخش بعضی از سطوح مربوط به این هرم را نمایش خواهیم داد از سطح صفر که برابر با تصویر اصلی است شروع خواهیم کرد و ادامه خواهیم داد:

سطح بعدی :

سطح بعدی :

سطح بعدی :

سطح بعدی :

سطح بعدی:

و به همین شکل ادامه خواهیم داد تا به یک پیسکل برسیم حال به سراغ بررسی مجموع تعداد پیکسل های تمامی سطوح تا رسیدن به یک پیکسل خواهیم رفت که برابر است با ۳۴۹۵۲۵ که برابر است با
4/3* (512) **2 است. و در بخش بعدی تصاویر مربوطه به سه سطح از هرم residual را نیز با همدیگر بررسی خواهیم کرد (بقیه سطوح به خاطر داشتن جزییات پایین آورده نشده اند و مانند همین سه سطح هستند فقط در اسکیل های کوچک تر)
سطح اول:

در این تصویر اگر با دقت بررسی کنید میتوانید ببینید که لبه های موجود در تصاویر قرار دارند سطح دوم:

در این تصویر نیز میتوان لبه های تصاویر را مشاهده کرد
سطح سوم:

لبه ها در این تصویر نیز قابل مشاهده هستند از این هرم برای `reconstruct` کردن استفاده خواهد شد.

۲-۳ تمرین ۲-۱-۶

در این تمرین از ما خواسته شده بود که در ابتدا هرم را با استفاده از فیلتر باکس فیلتر دو در دو ایجاد کنیم سپس با استفاده از `pixel replication` دوباره تصویر اصلی را ایجاد کنیم

ابتدا به سراغ بررسی هرم ساخته شده خواهیم رفت :

سطح صفر :

سطح یک:

سطح دو:

همانطور که مشخص است کیفیت تصویر نسبت به تصویر در سطح دوم هرم افت شدید کیفیت پیدا کرده است

سطح بالاتر:

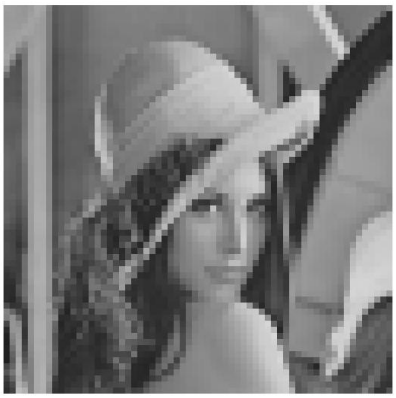
سطح سوم:

سپس با داشتن تصویر بالا که سطح سوم است سعی میکنیم دوباره تصویر اصلی را ایجاد کنیم و این کار را سطح به سطح انجام خواهیم داد :

سطح بالاتر :

سطح بالاتر:

سطح یک:



همانطور که مشخص این تصویر در مقایسه با تصویر اصلی و اولیه افت شدید کیفیت را دارد.

سطح دو:

۳-۳ تمرین ۳-۱-۶

در این تمرین از ما خواسته شده بود که با استفاده از Wavelet ترنسفورم با mother wavlet هار هرم را ایجاد کنیم که در ادامه تصاویر مربوط به هر یک از سطوح را بایکدیگر مشاهده خواهیم کرد :

سطح صفر :

سطح سوم:

همانطور که مشخص است تصاویر موجود در اسکیل های مختلف در این روش نسبت به روش قبلی که استفاده از فیلتر ۲ در ۲ میانگین گیر بود نتیجه به مقدار کمی بهتر است .

۴-۳ تمرین ۴-۱-۶

در این تمرین از ما خواسته شده است که فرمول داده شده به ما را بر روی تصویر اصلی اعمال کنیم و سپس `psnr` را برای تصویر خروجی نسبت به تصویر ورودی اعمال کنیم و نتیجه را گزارش کنیم در ابتدا با هم تصویر ورودی را بررسی خواهیم کرد :

حال به بررسی خروجی مربوطه بعد از اعمال فیلتر خواهیم پرداخت :

همانطور که مشخص است به خاطر اینکه مقدار `y = 2` است در نتیجه فرق در تصویر ایجاد نمیشود که به توان آن را با چشم مشاهده کرد اما اگر مقدار مربوط به هر یک از پیکسل ها را بررسی کنیم مشاهده خواهیم کرد که مقادیر با تصویر اصلی متفاوت است حال به سراغ سطح های بعدی خواهیم رفت:

که بعد از اعمال فیلتر به شکل زیر است:

برای سطح بعدی :

که بعد از اعمال فیلتر به شکل زیر خواهد شد:

و همچنین مقداری که تابع psnr به ما برخواهد گرداند در سطح اول برابر با ۴۸.۰۷۸۶ در سطح دوم برابر با منفی ۴۸ و در سطح سوم برابر با منفی ۵۴ است.

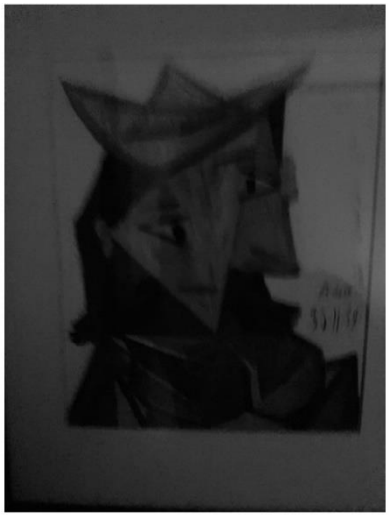
۵-۳ تمرین ۲-۶

در این تمرین از ما خواسته شده بود با استفاده از دو روش نویز موجود در تصاویر را حذف کنیم ابتدا تصاویر نویزی را باهم بررسی خواهیم کرد :

تصویر اول آلوده شده به نویز فلفل نمک :

تصویر دوم آلوده شده به نویز گوسی :

تصویر سوم از یک محیط طبیعی با نور کم :



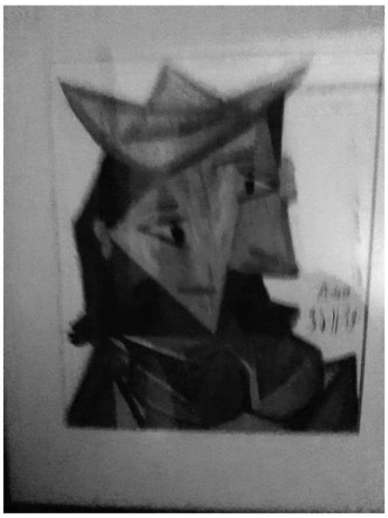
حال به بررسی اعمال فیلتر با hardtreshholding بر روی تصویر آغشته به نویز فلفل نمک خواهیم پرداخت :

همانطور که مشخص است نویز قدری کاهش یافته است حال به سراغ بررسی همین تصویر با اعمال Soft treshholding خواهیم رفت :

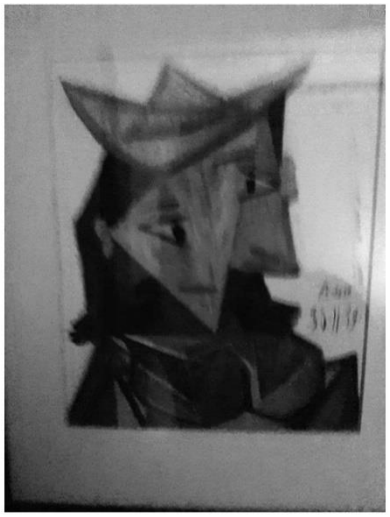
این تصویر نسبت به تصویر قبلی کیفیت بهتری دارد میتوان با افزایش مقدار treshhold نویز را کمتر کرد ولی ممکن است بر روی اجزای تصویر نیز اثر گذار باشد حال به سراغ بررسی اعمال hard treshholding بر روی تصویر آغشته به نویز گوسی خواهیم رفت :

همانطور که قابل مشاهده است قدری از نویز کم شده و کیفیت بهتر شده است حال به سراغ اعمال Soft treshholding بر روی تصویر آغشته به نویز گوسی خواهیم رفت :

این تصویر نسبت به تصویر قبلی کیفیت مطلوب تری دارد. حال به سراغ بررسی اعمال hard treshholding بر روی تصویر تهیه شده از محیط با نور کم خواهیم رفت :



همانطور که به وضوح قابل مشاهده است نویز به مقدار زیادی کم شده است و کیفیت بسیار بهتر شده است نسبت به تصویر اصلی حال به سراغ اعمال soft treshholding بر روی تصویر تهیه شده از محیط خواهیم رفت و آنرا بررسی خواهیم کرد:



در این روش نیز قابل مشاهده است که نویز به مقدار زیادی کاهش یافته است و کیفیت تصویر بهتر شده است.

۴-کد های مربوطه

۴-۱ تمرین ۱-۱-۶

```
clc;
clear all;
close all;

img = imread('Lena.bmp');
img = rgb2gray(img);
makePyramid(img,9);

%make the pyramid with guassian
filter
function [] =
makePyramid(img,level)
    imshow(img);
    [r,c] = size(img);
    counter = r*c;
    for i = 1 : level
        img2 = filter(img);
         res = img2 -
double(img);
        img =
downSampleByRemoval(img);
        [r,c] = size(img);
        counter = counter +
r*c;
        figure;
        imshow(uint8(img));
         figure;
            imshow(uint8(res));
    end
    disp(counter);
end
%down sampling by pixel removal
function [g] =
downSampleByRemoval(img)
    [r,c] = size(img);
    g = img(1:2:r,1:2:c);
end
%corrolate gussian filter with
image
function [outImg] = filter(img)
    [r,c] = size(img);
    outImg = zeros(r,c);
    padImg = zeros(r+2,c+2);
    padImg(2:r+1,2:c+1) = img;
    gussian = [1/16 1/8
1/16;1/8 1/4 1/8;1/16 1/8
1/16];
    for i = 2:r+1
        for j = 2:c+1
           outImg(i-1,j-1) =
sum(sum(double(padImg(i-
1:i+1,j-1:j+1)).*gussian));
        end
    end
end
```

۴-۲ تمرین ۲-۱-۶

```
clc;
clear all;
close all;

img = imread('Lena.bmp');
img = rgb2gray(img);
base = makePyramid(img,3);
reversePyramid(base,3);

%up sampling and reconstruction
function [] =
reversePyramid(img,level)
    for i = 1 : level
        img =
UpSampleByReplication(img);
        figure;
        imshow(uint8(img));
    end
end
%make pyramid
function [baseImg] =
makePyramid(img,level)
    imshow(img);
    for i = 1 : level
```

```
clc;
clear all;
close all;


img = imread('Lena.bmp');
img = rgb2gray(img);
%take three times wavelet
transform and show Low Low
image
[LL1, LH1, HL1, HH1] =
dwt2(im2double(img), 'haar');
figure;
imshow(mat2gray(LL1));
[LL2, LH2, HL2, HH2] =
dwt2(im2double(LL1), 'haar');
figure;
imshow(mat2gray(LL2));
[LL3, LH3, HL3, HH3] =
dwt2(im2double(LL2), 'haar');
figure;
imshow(mat2gray(LL3));
```

۴-۴ تمرین ۴-۱-۶

```
clc;
clear all;
close all;

img = imread('Lena.bmp');
img = rgb2gray(img);
%apply wavelet trnsform
[LL, LH, HL, HH] =
dwt2(im2double(img), 'haar');
%quantize the image
LL1 = quantize(LL,2);
LH1 = quantize(LH,2);
HL1 = quantize(HL,2);
HH1 = quantize(HH,2);
%apply inverse wavelet
transform
output =
idwt2(LL1,LH1,HL1,HH1,'haar');
imshow(uint8(output));
disp(psnr(img,uint8(output)));
figure;
imshow(img);

%quantization function
function [g] = quantize(f,y)
    f = f.*255;
    [r,c] = size(f);
    g = zeros(r,c);
        img = filter(img);
        img =
downSampleByRemoval(img);
        figure;
        imshow(uint8(img));
        baseImg = img;
    end
end
%down sampling image by pixel
removal
function [g] =
downSampleByRemoval(img)
    [r,c] = size(img);
    g = img(1:2:r,1:2:c);
end
%correlate box filter with
image
function [outImg] = filter(img)
    [r,c] = size(img);
    outImg = zeros(r,c);
    padImg = zeros(r+2,c+2);
    padImg(2:r+1,2:c+1) = img;
    boxFilter = [1/4 1/4;1/4
1/4];
    for i = 2:r+1
        for j = 2:c+1
            outImg(i-1,j-1) =
sum(sum(double(padImg(i:i+1,j:j
+1)).*boxFilter));
        end
    end
end
%up sampling function using
pixel replication
function [g] =
UpSampleByReplication(img)
   [r,c] = size(img);
   g = zeros(r*2,c*2);
   for i=1:2:2*r
       for j=1:2:2*c
           g(i,j) =
img(ceil(i/2),ceil(j/2));
           g(i+1,j) =
img(ceil(i/2),ceil(j/2));
           g(i,j+1) =
img(ceil(i/2),ceil(j/2));
           g(i+1,j+1) =
img(ceil(i/2),ceil(j/2));
       end
   end
end
```

۴-۳ تمرین ۳-۱-۶

```
out3 =
denoise(LL2,LH2,HL2,HH2,'hard',
trsh);
figure;
imshow(mat2gray(out3));
out4 =
denoise(LL2,LH2,HL2,HH2,'soft',
trsh);
figure;
imshow(mat2gray(out4));
out5 =
denoise(LL3,LH3,HL3,HH3,'hard',
trsh);
figure;
imshow(mat2gray(out5));
out6 =
denoise(LL3,LH3,HL3,HH3,'soft',
trsh);
figure;
imshow(mat2gray(out6));

%function for adding treshhold
constraints and take inverse
wavelet
function[g] =
denoise(LL,LH,HL,HH,treshhold,t
resh)
    if treshhold == 'hard'
        lh =
hardTreshhold(LH,tresh);
        hl =
hardTreshhold(HL,tresh);
        hh =
hardTreshhold(HH,tresh);
        g =
idwt2(LL,lh,hl,hh,'haar');
    elseif treshhold == 'soft'
        lh =
softTreshhold(LH,tresh);
        hl =
softTreshhold(HL,tresh);
        hh =
softTreshhold(HH,tresh);
        g =
idwt2(LL,lh,hl,hh,'haar');
    end
end
%hard treshhloding function for
LH,HL,HH
function[g] =
hardTreshhold(f,tresh)
    [r,c] = size(f);
    g = zeros(r,c);
    for i=1:r
        for j=1:c
            if f(i,j)<tresh
                for i=1:r
                    for j = 1 :c
                        g(i,j) = y *
sign(f(i,j)) *
floor(abs(f(i,j))/y);
                    end
                end
end
```

۵-۴ تمرین ۲-۶

```
clc;
clear all;
close all;

img = imread('Lena.bmp');
img = rgb2gray(img);
testImg = imread('test.jpg');
testImg = rgb2gray(testImg);
%add noise to the image
saltandpeper =
imnoise(img,'salt & pepper');
gaussian =
imnoise(img,'gaussian',0.05);
imshow(saltandpeper);
figure;
imshow(gaussian);
figure;
imshow(testImg);

%wavelet transform
[LL, LH, HL, HH] =
dwt2(im2double(saltandpeper),
'haar');
[LL2, LH2, HL2, HH2] =
dwt2(im2double(gaussian),
'haar');
[LL3, LH3, HL3, HH3] =
dwt2(im2double(testImg),
'haar');

%donoising
trsh = 0.8;
out1 =
denoise(LL,LH,HL,HH,'hard',trsh
);
figure;
imshow(mat2gray(out1));
out2 =
denoise(LL,LH,HL,HH,'soft',trsh
);
figure;
imshow(mat2gray(out2));
```

```
                g(i,j) = 0;
            else
                g(i,j) =
f(i,j);
            end
        end
    end
end
%Soft treshhloding function for
LH,HL,HH
function[g] =
softTreshhold(f,tresh)
    [r,c] = size(f);
    g = zeros(r,c);
    for i=1:r
        for j=1:c
            if f(i,j)<tresh
                g(i,j) = 0;
            else
                g(i,j) =
f(i,j)-tresh;
            end
        end
    end
end
```