تمرین ۳

هادی تمیمی
9622762408

اطلاعات گزارش

تاریخ: ۱۴۰۰/۲/۶

چکیده

در این تمرین با فیلترهای مختلف و حذف نویز میپردازیم.

واژگان کلیدی:
فیلترینگ مکانی
فیلتر جعبه
ماسکینگ غیرشارپ
فیلتر میانگین
فیلتر میانه
شناسایی لبه
نویز نمک و فلفل
نویز گوسین

۱-مقدمه

فیلترینگ مکانی یک تصویر را به این صورت تغییر میدهد که مقدار جدید هر پیکسل به کمک یک تابع از مقدار آن پیکسل و همسایه های آن بدست می آید.
اگر عملیاتی که بر روی پیکسل اجرا میشود خطی باشد، فیلتر را فیلتر مکانی خطی می‌نامیم.در غیر اینصورت آن را فیلتر مکانی غیرخطی گوییم.
یک فیلتر مکانی خطی عملیات جمعِ ضرب ها را بین تصویر f و فیلتر w انجام میدهد.ضرایب فیلترمشخص کننده خواص فیلتر می‌باشد.
به این فیلتر ، ماسک یا پنجره می‌گویند.

۲-شرح تکنیکال

۳.۱ فیلتر جعبه

این فیلتر،یک فیلتر خطی است. اگر فیلتر جعبه $\boldsymbol{n} \times \boldsymbol{n}$ داشته باشیم، جمع مقادیر پکسل ها در پنجره ای به طول n پیکسل به مضرب $\frac{\mathbf{1}}{\boldsymbol{n^2}}$ برای تعیین پیکسل مرکزی آن پنجره است.

۳.۱.۱

یکی از مشکلات این فیلتر حساس بودن آن به نویز میباشد.زیرا نویز معمولا با پیکسل های اطرافش اختلاف مقدار زیادی دارد و در میانگین گیری میتواند تاثیر منفی زیادی داشته باشد.

۳.۱.۲

ویژگی های منفی در ابتدا زیاد میشوند اما به مرور با تکرار فیلتر تصویر تغییری پیدا نمیکند.دلیل این اتفاق قضیه آماری central limit میباشد و با تکرار فیلتر سطح خاکستری پیکسل های تصویر به سمت توزیع نرمال همگرا میشود.

۳.۱.۳

با تکرار فیلتر برروی تصویر،تصویر blur تر میشود .زیرا با میانگین گیری پشت سر هر پیکسل بیشتر در تاثیر میانگین پیکسل های اطرافش قرار میگیرد

۳.۱.۴

با انتخاب سایز ماسک بزرگ، نویز بیشتر گرفته میشود اما همزمان تصویر blur تر میشود .زیرا با انتخاب ماسک بزگتر پیکسل های بیشتر در میانگین گیری انتخاب میشوند و هر پیکسل تحت تاثیر تعداد بیشتری از پیکسل های اطرافش قرار میگیرد.

۳.۱.۵

انتخاب سایز ماسک برای هر تصویر متفاوت است باید به نسبت نویز موجود در تصویر انتخاب شود. در این تصویر انتخاب سایز ماسک ۳*۳ نتیجه قابل قبولی ایجاد می شود. در صورت اجرای فیلتر بر روی تصویر اصلی Elaine تصویر حاصل از اجرای یکبار فیلتر تصویر تا حدی قابل تشخیص است اما پس از تکرار فیلتر تصویر روی به سیاه شدن میرود.

۳.۱.۶

با اجرای این مسئله رو خروجی ۳.۱.۳ (تصویر حاصل از فیلتر بیستم) پس از هر بار اعمال فیلتر تیره تر میشود تا جایی که تصویر کامل سیاه می شود.

۳.۲ فیلتر میانه

این فیلتر،یک فیلتر غیرخطی است.در یک پنجره مشخص برای یک پیکسل خاص،میانه مقادیر پیکسل های درون پنجره را به عنوان مقدار جدید پیکسل در نظر میگیریم.

۳.۲.۱

در نتیجه فیلتر میانه بر روی نویز فلفل و نمک عملکرد خوبی دارد. برای انتخاب سایز پنچره مناسب بر اساس میزان density نویز میتوان تصمیم گرفت. اگر نویز در تصویر چگالی زیادی داشته باشد،با سایز پنجره کم ممکن است تصویر بطور کامل خراب شود و سایز پنجره بزرگ میتواند تصویر را بیش از حد blur کند در صورتی که با انتخاب پنجره کوک تر و مناسب هم نویز را از بین برد و هم حداقل میزان blur شدن را داشت.. برای همین برای هر تصویر باید متناسب با مقدار نویز سایز پنجره مناسب خود انتخاب شود.

۳.۲.۲

فیلتر میانگین بر روی تصاویر با نویز عملکرد ضعیفی دارد. هر چقدر سایز فیلتر بزرگتر باشد،این ضعف عمیق تر میشود .هرچقدر فیلتر میانگین بزرگتری داشته باشید، این شدت تغییرات بیشتر خواهد بود. از سوی دیگر فیلتر میانه بر

روی نـویز گاوسـین عملکـرد قابـل قبـولی دارد امـا اگـر انـدازه پنجـره آن بـیش از حـد بـزرگ انتخـاب شـود نیـز تصـویر حاصل blur میشود.

۳.۲.۳

بـا اسـتفاده از فیلتـر median میتـوان بـه خـوبی نیـز نمـک و فلفـل و گوسـین را گرفـت. ابتـدا بـا انتخـاب انـدازه پنجـره هـای متفـاوت(از کوچـک بـه بـزرگ در بـازه محـدودی) و مقایسـه نتیجـه هـای MSE(orginal_img,filtered_img) پنجـره ای کـه کمتـرین مقـدار نتیجـه MSE را داشـت را بـه عنـوان پنجـره ی فیلتـر median بـرای آن تصـویر انتخـاب میکنیم(تا بهترین تعـادل بـین حـذف نـویز و تـاری تصـویر انتخـاب شـود). کـه در مثـال تصـویر Elaine بـا نـویز گوسـین 0.05 و نویز نمک و فلفل 0.1 ، در محدوه پنجره ی [7 .. 2] پنجره ی ۵*۵ بهترین نتیجه را میدهد .

۳.۳

۳.۳.۱

با داشتن تصویری تاریک و blur و نیوز شبه فلفل و نمک و فیلتر ابتدا با فیلتر میانه نویز فلفل و نمک را حذف میکنیم و سپس با مقادیر مناسب فیلتر unsharp masking (high boost) تصویر را sharp می کنیم و با افزایش مقدار هر پیکسل تصویر را روشن تر می کنیم.

۳.۴

۳.۴.۱

این سه فیلتر لبه های عمودی تصویر را شناسایی میکنند. فیلتر b بسیار شبیه به فیلتر c است با این تفاوت که اهمیت کمتری پیکسل های کناری پیکسل مرکزی میدهد.در نتیجه شدت لبه کمتر میباشد.

فیلتر a با دو فیلتر دیگر تفاوت دارد.در واقع این فیلتر لبه های بیشتری را شناسایی میکند حتی اگر در واقعیت لبه خاصی در جایی که این فیلتر شناسایی کرده وجود نداشته باشد.زیرا این فیلتر برای فعال شدن فقط دو پیکسل کناری را در نظر میگیرد در حالی که دو فیلتر دیگر ۴ پیکسل مورب را نیز در نظر میگیرند و زمانی که لبه ای را پیدا کنند با اطمینان خاطر بیشتری میتوان گفت که آن واقعا یک لبه است.

فیلتر اول به دو فیلتر دیگر بار محاسباتی کمتری دارد و استفاده از آن بهینه تر است.

۳.۴.۲

فیلتر a تخمین مشتق افقی را محاسبه کرده و لبه های افقی را پیدا میکند. فیلتر b تخمین مشتق عمودی را محاسبه کرده و لبه های عمودی را پیدا میکند.

تفاوت این فیلترها با فیلترهای بخش قبل در این است که فیلترهای روبرت لبه های مورب و جزئی بیشتری را شناسایی میکنند.ولی فیلترهای b و c بخش قبل جزئیات کلی لبه ها را بدست می آورند.در مورد حجم محاسبات فیلترهای روبرت حجم محسباتب کمتری نسبت به فیلتر bو c بخش قبل دارد.

۳.۵

۳.۵.۱

یک فیلتر ماسکینگ غیرشارپ ساده به فرم زیر میباشد:

$$(1 - \alpha)I + \alpha I' = I + \alpha(I' - I)$$

کـه بـرای شـارپ کـردن تصـویر مـی توانـد اسـتفاده شـود کـه در آن I تصـویر اصـلی، 'I تصـویر smooth شـده اسـت و α را بـه صورت دستی انتخاب مـی شـود.در صـورتی کـه مقـدار α زیـاد باشـد،تاثیر تصـویر هموارشـده بـر روی نتیجـه بیشـتر میشـود و هرچـه

مقـدار آن کمتـر باشد،تصـویر اصـلی بیشـترین تـاثیر را بـر روی نتیجـه خواهـد گذاشـت. در صـورت انتخـاب مقـادیر زیـاد لبـه هـای تصویر برجسته تر میشود.

۲-شرح نتایج

۳.۱.۳

۵ تکرار

۱۵تکرار

۳۰تکرار

۴۰تکرار

۳.۱.۴

پنجره ۳*۳

پنجره ۷*۷

پنجره ۱۱*۱۱

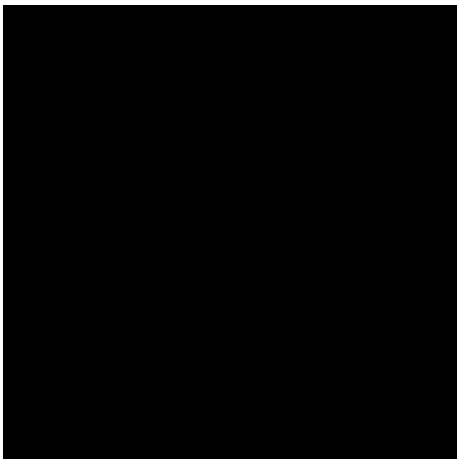
پنجره ۱۸*۱۸

۳.۱.۵

تصویر اصلی

فیلتر میانگین ۳*۳

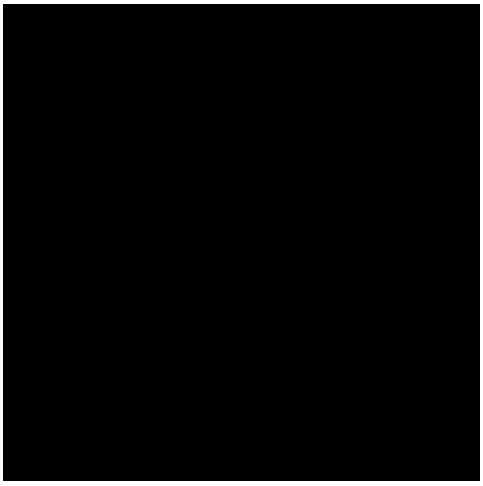
۳.۱.۶



۵ بار تکرار

۱ بار فیلتر

۱ بار فیلتر روی تصویر اصلی Eliane



۱۵ بار فیلتر روی تصویر اصلی Eliane

۳.۲.۱

MSE(orginal img , filtered img)

MSE	$3*3$	$5*5$	$7*7$	$9*9$	11*11
$\rho = 0.05$	۵۵٫۲۰۱	۴۵٫۵۱	۵۵٫۲۰۱	۷۰٫۷۴۲	۸۷٫۸۴۴
$\rho = 0.1$	۴۵٫۴۳۶	۵۳٫۱۷۷	۶۳٫۲۴۴	۸۲٫۲۱۱	۱۰۶٫۰۳
$\rho = 0.2$	۹۳٫۰۴	۸۵٫۷۵۶	۱۱۳٫۹	۱۵۰٫۹۲	۱۹۷٫۴۵
$\rho = 0.4$	۸۴۸٫۱۸	۱۸۳٫۸۶	۲۱۷٫۱۵	۲۶۶٫۸	۳۱۶٫۱
$\rho = 0.5$	۱۹۴۷٫۹	۳۵۳٫۶۷	۲۹۳٫۲۹	۳۴۰٫۹۳	۴۱۷٫۰۳

تصویر نویز دار شده $\rho = 0.05$

تصویر با فیلتر۳*۳ و $\rho = 0.05$

تصویر با فیلتر ۵*۵ و ρ = 0.05

تصویر با فیلتر ۷*۷ و ρ = 0.05

تصویر با فیلتر ۷*۷ و ρ = 0.05

تصویر با فیلتر ۱۱*۱۱ و ρ = 0.05

تصویر نویز دار شده ρ = 0.1

تصویر با فیلتر۳*۳ و ρ = 0.1

تصویر با فیلتر۵*۵ و $\rho = 0.1$

تصویر با فیلتر۷*۷ و $\rho = 0.1$

تصویر با فیلتر۹*۹ و $\rho = 0.1$

تصویر با فیلتر۱۱*۱۱ و $\rho = 0.1$

تصویر نویز دار شده $\rho = 0.2$

تصویر با فیلتر۳*۳ و $\rho = 0.2$

تصویر با فیلتر۵*۵ و ρ = 0.2

تصویر با فیلتر۷*۷ و ρ = 0.2

تصویر با فیلتر۹*۹ و ρ = 0.2

تصویر با فیلتر۱۱*۱۱ و ρ = 0.2

تصویر نویز دار شده ρ = 0.5

تصویر با فیلتر۳*۳ و ρ = 0.5

تصویر با فیلتر۵*۵ و $\rho = 0.5$

تصویر با فیلتر۷*۷ و $\rho = 0.5$

تصویر با فیلتر۹*۹ و $\rho = 0.5$

تصویر با فیلتر۱۱*۱۱ و $\rho = 0.5$

۳.۲.۲

MSE(orginal img , filtered img)

median	$3 * 3$	$5 * 5$	$7 * 7$	9*9	11*11
$\rho = 0.01$	۱۵۳٫۴۷	۱۰۲٫۲۵	۹۸٫۶۱۹	۱۱۲٫۴۳	۱۳۲٫۹
$\rho = 0.05$	۳۰۶٫۶۲	۲۵۳٫۹۴	۲۴۷٫۷۶	۲۵۸٫۹۹	۲۷۶٫۶۱
$\rho = 0.1$	۷۸۰٫۸۹	۷۲۹٫۴۱	۷۲۱٫۲۸	۷۳۰٫۱۲	۷۴۵٫۴۶

Box filter	$3 * 3$	$5 * 5$	$7 * 7$	9*9	11*11
$\rho = 0.01$	۵۵۵٫۶۴	۸۲۷٫۷۲	۱۱۳۹٫۳	۱۴۶۰٫۱	۱۷۸۵٫۶
$\rho = 0.05$	۷۰۱٫۰۲	۹۷۱٫۵۵	۱۲۸۱٫۱	۱۵۹۹٫۲	۱۹۲۲٫۵
$\rho = 0.1$	۱۱۵۲٫۵	۱۴۱۶٫۸	۱۷۱۹٫۶	۲۰۳۱٫۶	۲۳۴۸٫۶

فیلتر میانگین ۷*۷, $\rho = 0.05$

فیلتر میانه ۷*۷ , $\rho = 0.05$

۳.۲.۳

median	$2 * 2$	3*3	4*4	$5 * 5$	6*6	7*7	7*7
	۶۴۸٫۴۱	۳۵۱٫۶۵	۳۵۸٫۳۲	۲۷۳٫۶۵	۳۴۳٫۸۲	۱۳۲٫۹	۲۶۲٫۱۱

نویز گوسین 0.05 و نویز نمک و فلفل 0.1

تصویر با فیلتر ۵*۵ میانه

۳.۳.۱

تصویر نویز دار در تاریکی

تصویر پس از فیلتر و شارپ شدن و روشن شدن

۳.۴.۱

حاصل اعمال فیلتر **a**

حاصل اعمال فیلتر **b**

حاصل اعمال فیلتر **c**

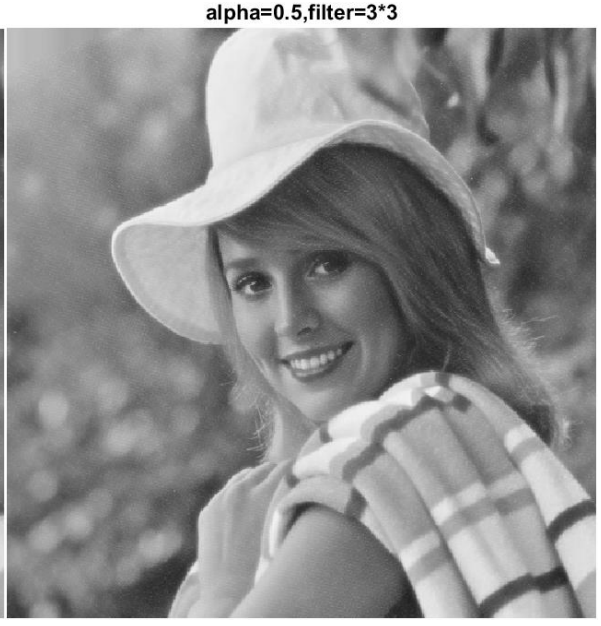
۳.۴.۲

حاصل اعمال فیلتر **a**

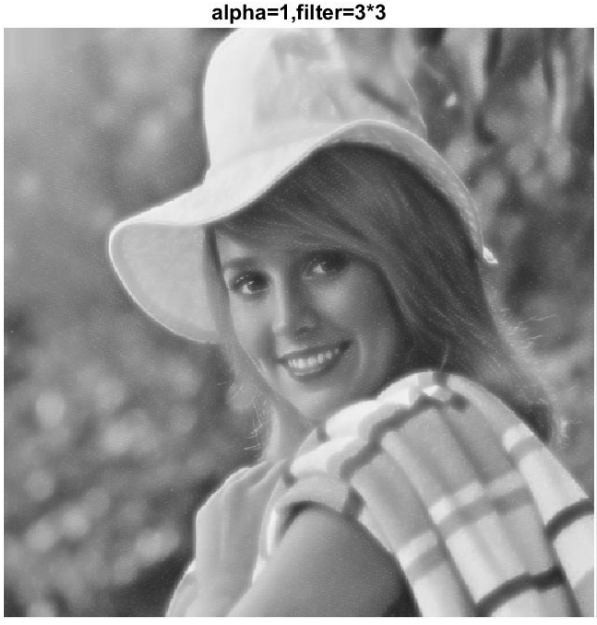
حاصل اعمال فیلتر **b**

۳.۵.۱

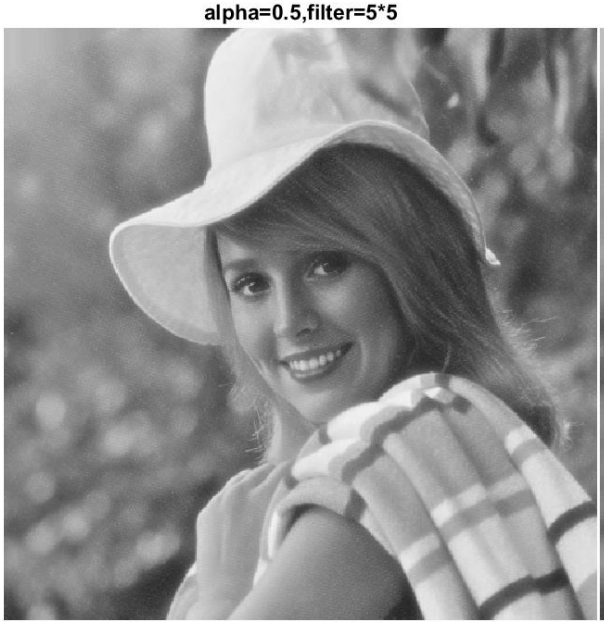
alpha=0.5,filter=3*3



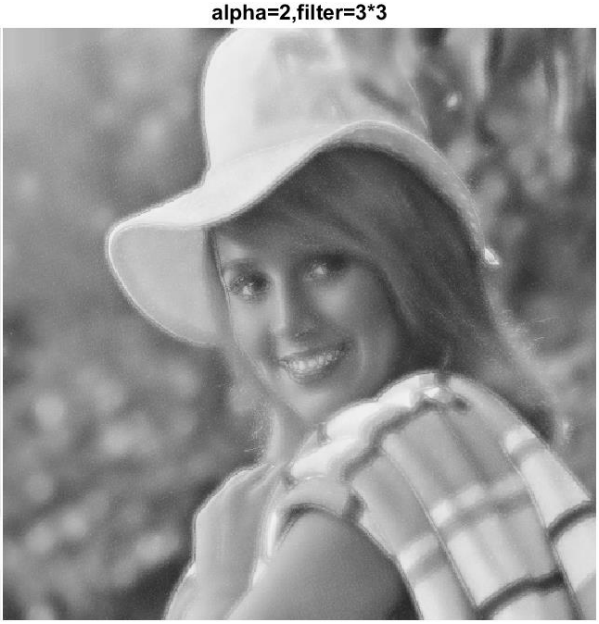
alpha=1,filter=3*3



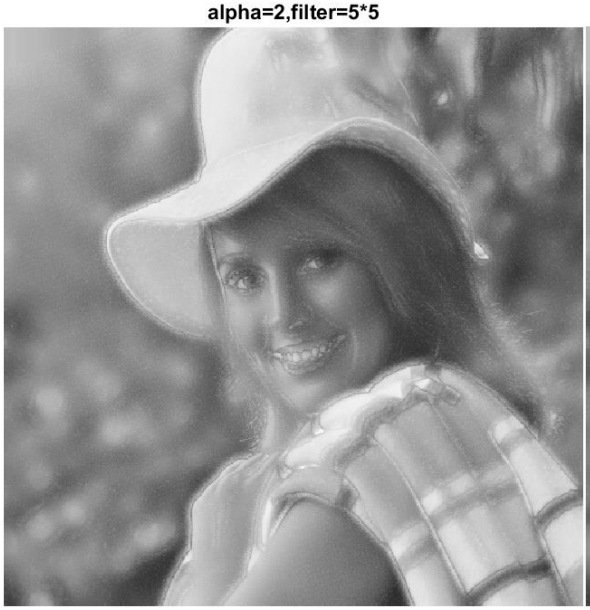
alpha=0.5,filter=5*5



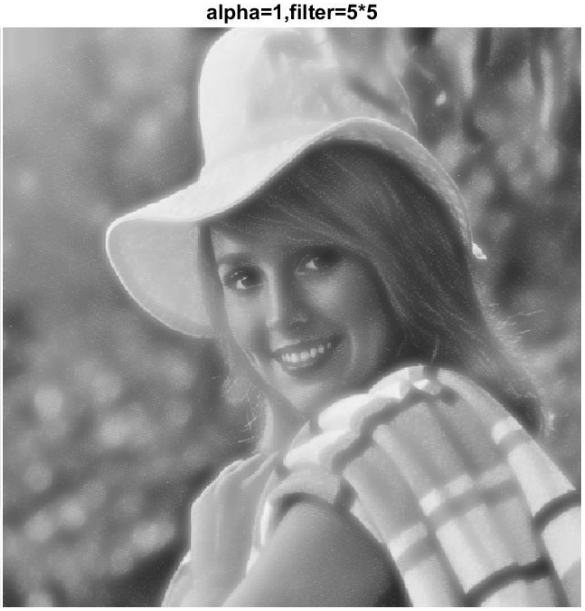
alpha=2,filter=3*3



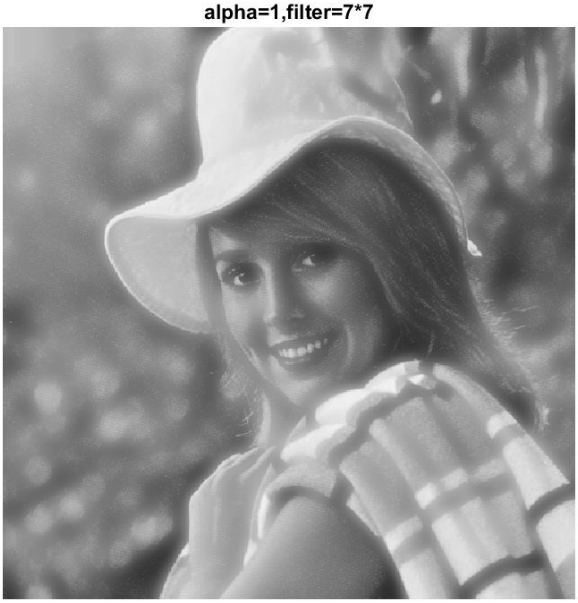
alpha=2,filter=5*5



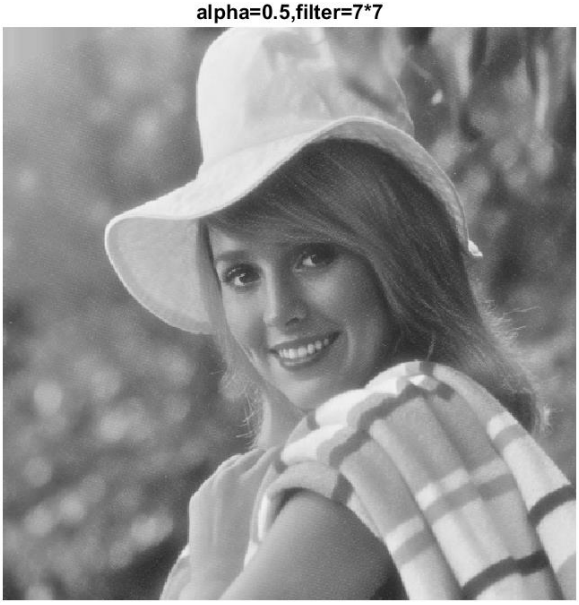
alpha=1,filter=5*5



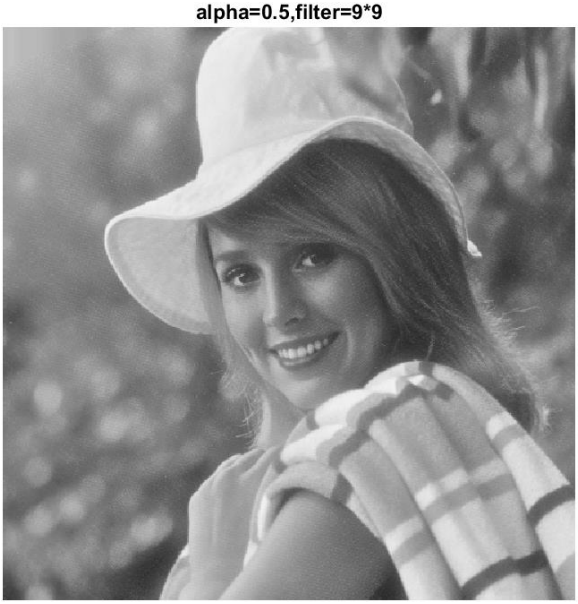
alpha=1,filter=7*7



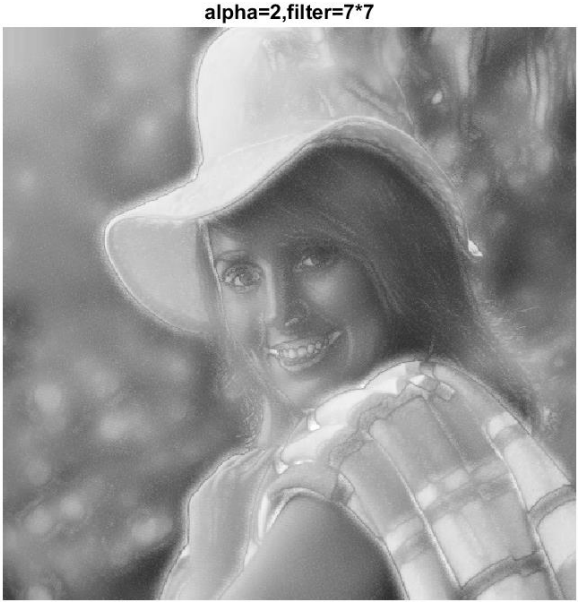
alpha=0.5,filter=7*7



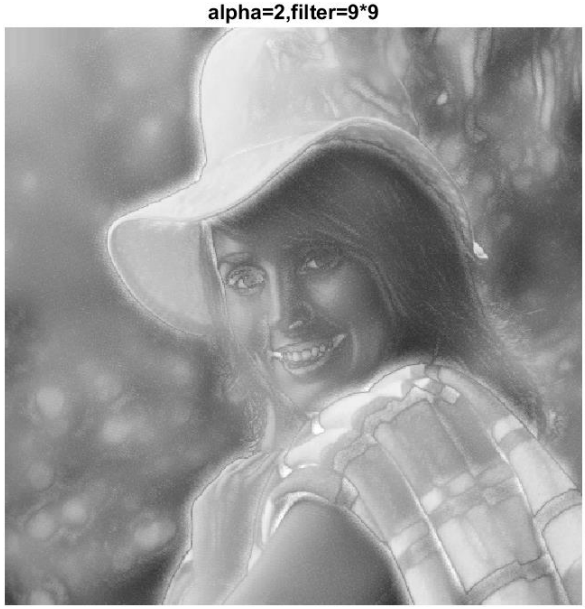
alpha=0.5,filter=9*9



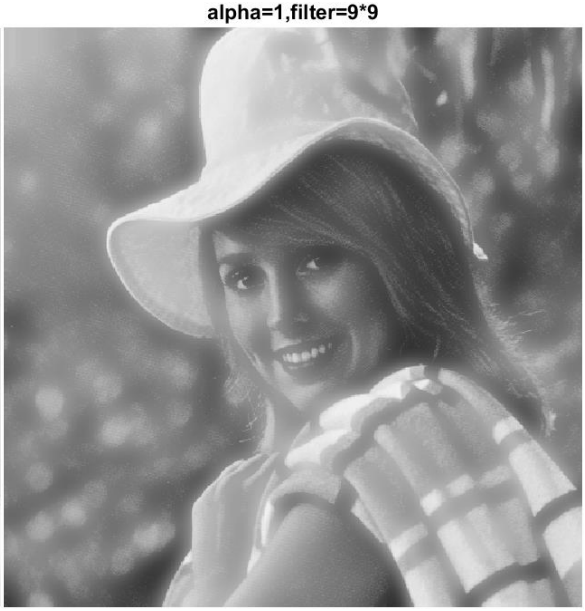
alpha=2,filter=7*7



alpha=2,filter=9*9

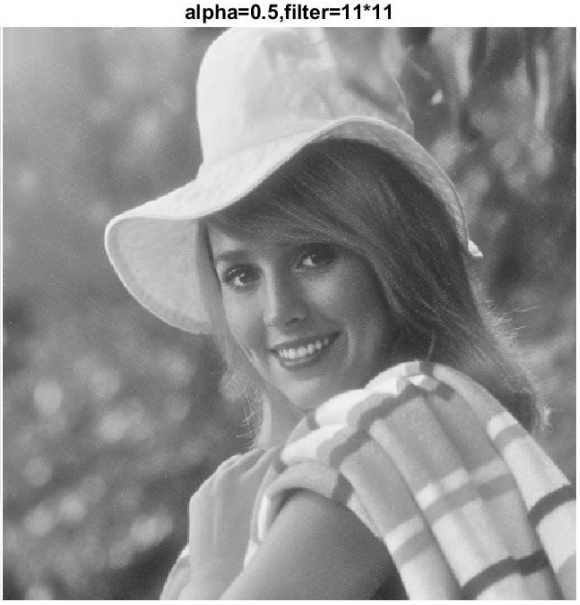


alpha=1,filter=9*9

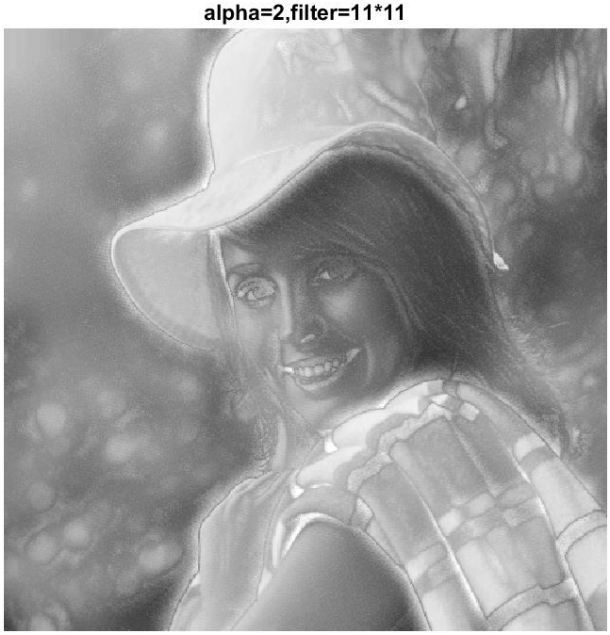


alpha=1,filter=11*11

alpha=0.5,filter=11*11



alpha=2,filter=11*11



کد ها:

۳.۱.۳

لود کردن تصویر و فراخوانی توابع و نمایش خروجی:

```
image = imread('Images/3/Elaine.bmp');
imwrite(image,'ORGINAL.png');
```

```
filter_size =3;
box_filter  = ones(filter_size,filter_size); %creates 3*3 box_filter
box_filter = uint8(box_filter);

output = box_filtering(image,box_filter,filter_size);
step = 5;
max_range=50;
for i=1:max_range
    if(mod(i,step)==0)
    imwrite(output,"FILTER("+i+").png");
    end
    output = box_filtering(output,box_filter,filter_size);
end
```

تابع box filtering:

```
function output=box_filtering(img,filter,filter_size)
    [R,C] = size(img);
    img = padarray(img,[1,1]);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C
            if(i-filter_size>=0 && j-filter_size>=0 &&
i+filter_size<=R && j+filter_size<=C)
            part = img(i:(i+filter_size-1),j:(j+filter_size-1));
            mult = part.*filter;
            out = sum(mult,'all')/(filter_size^2);
            output(i,j) = out;
            end
        end
    end
end
```

۳.۱.۴

لود کردن تصویر و فراخوانی توابع و نمایش خروجی:

```
image = imread('Images/3/Elaine.bmp');
imwrite(image,'ORGINAL.png');
filter_size = [3;7;11;18]; %change this for difrent mask size

for i=1:size(filter_size)
    box_filter  = ones(filter_size(i),filter_size(i)); %creates X*X
box_filter
    box_filter = uint8(box_filter);
    output = box_filtering(image,box_filter,filter_size(i));
    imwrite(output,"FILTER SIZE("+filter_size(i)+").png");
end
```

تابع box filtering:

```
function output=box_filtering(img,filter,filter_size)
    [R,C] = size(img);
    x= floor(filter_size/2-1);
    img = padarray(img,[x,x]);
    output = zeros(R,C,'uint8');
    for i=1:R
```

```
        for j=1:C
            if(i-filter_size>=0 && j-filter_size>=0 &&
i+filter_size<=R && j+filter_size<=C)
            part = img(i:(i+filter_size-1),j:(j+filter_size-1));
            mult = part.*filter;
            out = sum(mult,'all')/(filter_size^2);
            output(i,j) = out;
            end
        end
    end
```

۳.۱.۵

فراخوانی توابع و نمایش خروجی:

```
image = imread('Images/3/Elaine.bmp');
imwrite(image,'ORGINAL.png');

for filter_size=2:6  %mask size 3 have a good balance
box_filter  = ones(filter_size,filter_size); %creates X*X box_filter
box_filter = uint8(box_filter);
output = box_filtering(image,box_filter,filter_size);
imwrite(output,"FILTER SIZE("+filter_size+").png");
end
```

تابع box filtering:

```
function output=box_filtering(img,filter,filter_size)
    [R,C] = size(img);
    x= floor(filter_size/2-1);
    img = padarray(img,[x,x]);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C
            if(i-filter_size>=0 && j-filter_size>=0 &&
i+filter_size<=R && j+filter_size<=C)
            part = img(i:(i+filter_size-1),j:(j+filter_size-1));
            mult = part.*filter;
            out = sum(mult,'all')/(filter_size^2);
            output(i,j) = out;
            end
        end
    end
end
```

۳.۱.۶

لود کردن تصویر و فراخوانی توابع و نمایش خروجی:

```
%image = imread('Images/3/Elaine.bmp');
image = imread('Images/3/FILTER(20).png');
imwrite(image,'ORGINAL.png');

filter_size =3;
box_filter=[0 -1 0 ; -1 5 -1 ; 0 -1 0]; %creates 3*3 box_filter
box_filter = uint8(box_filter);
```

```
output = box_filtering(image,box_filter,filter_size);
max_range=50;
for i=1:max_range
    imwrite(output,"FILTER("+i+").png");
    output = box_filtering(output,box_filter,filter_size);
end
```

تابع box filtering:

```
function output=box_filtering(img,filter,filter_size)
    [R,C] = size(img);
    img = padarray(img,[1,1]);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C
            if(i-filter_size>=0 && j-filter_size>=0 &&
i+filter_size<=R && j+filter_size<=C)
            part = img(i:(i+filter_size-1),j:(j+filter_size-1));
            mult = part.*filter;
            out = sum(mult,'all')/(filter_size^2);
            output(i,j) = out;
            end
        end
    end
end
```

۳.۲.۱

لود کردن تصویر و ایجاد نویز و فراخوانی توابع و نمایش خروجی:

```
image = imread('Images/3/Elaine.bmp');
format shortg
imwrite(image,"orginal.png");
% change comment for diffrent noise ****
%noised_img = imnoise(image,'salt & pepper',0.05);
%noised_img = imnoise(image,'salt & pepper',0.1);
%noised_img = imnoise(image,'salt & pepper',0.2);
%noised_img = imnoise(image,'salt & pepper',0.4);
noised_img = imnoise(image,'salt & pepper',0.5);

imwrite(noised_img,"noised.png");
filters = [3;5;7;9;11];
filter_numbers = size(filters);
%immses = zeros(4,1,'double');
%immses=[filter_numbers];
for i=1:filter_numbers
        filtered_img=median_filter(noised_img,filters(i));
        imwrite(filtered_img,"filter"+filters(i)+".png");
        %immse(image,x);
        %immses(i) = immse(image,x);
        mse_ans(i) = immse(image,filtered_img);


end
mse_ans %showing mse table
```

تابع median filtering:

```
function output=median_filter(image,length)
    [R,C] = size(image);
    image = padarray(image,[floor(length/2),floor(length/2)]);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C
            part = image(i:i+length-1,j:j+length-1);
            out = median(part,'all');
            output(i,j) = out;
        end
    end
end
```

۳.۲.۲

فراخوانی توابع و ایجاد نویز و اجرای فیلتر های متفاوت:

```
image = imread('Images/3/Elaine.bmp');
imwrite(image,"orginal.png");
format shortg

%change comment for diffrent input

%noised_img = imnoise(image,'gaussian',0.01);
noised_img = imnoise(image,'gaussian',0.05);
%noised_img = imnoise(image,'gaussian',0.1);

imwrite(noised_img,"noised.png");
filters = [3;5;7;9;11];
filter_numbers = size(filters);
    for i=1:filter_numbers
        box_filter  = ones(filters(i),filters(i)); %creates X*X
box_filter
        box_filter = uint8(box_filter);
        avg_filterd_img =
box_filtering(noised_img,box_filter,filters(i));
        imwrite(avg_filterd_img,"avg_filter"+i+".png");
        avg_immse_ans(i) = immse(image,avg_filterd_img);
    end
disp('avg filter')
avg_immse_ans %showing mse table


    for j=1:filter_numbers

        median_filterd_img =
median_filtering(noised_img,filters(j));
        imwrite(median_filterd_img,"median_filter"+j+".png");
        median_immse_ans(j) = immse(image,median_filterd_img);

    end
disp('median filter')
median_immse_ans %showing mse table
```

تابع box filtering:

```
function output=box_filtering(img,filter,filter_size)
    [R,C] = size(img);
    x= floor(filter_size/2-1);
    img = padarray(img,[x,x]);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C
            if(i-filter_size>=0 && j-filter_size>=0 &&
i+filter_size<=R && j+filter_size<=C)
            part = img(i:(i+filter_size-1),j:(j+filter_size-1));
            mult = part.*filter;
            out = sum(mult,'all')/(filter_size^2);
            output(i,j) = out;
            end
        end
    end
end
```

تابع median filtering:

```
function output=median_filtering(image,window_size)
    [R,C] = size(image);
    image =
padarray(image,[floor(window_size/2),floor(window_size/2)]);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C
            part = image(i:i+window_size-1,j:j+window_size-1);
            out = median(part,'all');
            output(i,j) = out;
        end
    end
end
```

۳.۲.۳

لود کردن تصویر و فراخوانی توابع و اجرای فیلتر های متفاوت و نمایش خروجی:

```
image = imread('Images/3/Elaine.bmp');
imwrite(image,"orginal.png");
format shortg
gaussian = imnoise(image,'gaussian',0.05);
noised_image = imnoise(gaussian,'salt & pepper',0.1);
imwrite(noised_image,"noised.png");

filters = [2;3;4;5;6;7];
filter_numbers = size(filters);
for i=1:filter_numbers
    median_filterd_img = median_filtering(noised_image,filters(i));
    imwrite(median_filterd_img,"median("+filters(i)+").png");
    median_immse_ans(i) = immse(image,median_filterd_img);
end
median_immse_ans % shows MSE result
```

تابع median filtering:

```
function output=median_filtering(image,window_size)
    [R,C] = size(image);
    image =
padarray(image,[floor(window_size/2),floor(window_size/2)]);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C
            part = image(i:i+window_size-1,j:j+window_size-1);

            out = median(part,'all');
            output(i,j) = out;
        end
    end
end
```

۳.۳.۱

لود کردن تصویر و فراخوانی توابع و نمایش خروجی:

```
image = imread('Images/3/noisy2.jpg');
image = rgb2gray(image);

x = median_filtering(image,7);
x= unsharp_masking_filter(x,2,4); % sharping
x=brightness(x,10); % increase bightness

imwrite(image,"orginal.png");
imwrite(x,"out.png");
```

تابع unsharp masking:

```
function output=unsharp_masking_filter(image,alpha,i)
        smoothed_img = imgaussfilt(image,i);
        output = image + alpha*(smoothed_img - image);
end
```

تابع تغییر brightness:

```
function output=brightness(img,add)
    [R,C] = size(img);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C

            img(i,j)=img(i,j)+add;

            output(i,j) = img(i,j);
        end
    end
end
```

تابع median filtering:

```
function output=median_filtering(image,window_size)
    [R,C] = size(image);
```

```
    image =
padarray(image,[floor(window_size/2),floor(window_size/2)]);
    output = zeros(R,C,'uint8');
    for i=1:R
        for j=1:C
            part = image(i:i+window_size-1,j:j+window_size-1);
            out = median(part,'all');
            output(i,j) = out;
        end
    end
end
```

۳.۴.۱

لود کردن تصویر و ایجاد فیلتر ها و فراخوانی توابع و نمایش خروجی:

```
image = imread('Images/3/Elaine.bmp');
image = double(image);

filter1 = [1 0 -1]/2;
filter2 = [1 0 -1;1 0 -1;1 0 -1]/6;
filter3 = [1 0 -1;2 0 -2;1 0 -1]/8;

img1 = filtering(image,filter1);
img2 = filtering(image,filter2);
img3 = filtering(image,filter3);

imwrite(img1,'out1.png');
imwrite(img2,'out2.png');
imwrite(img3,'out3.png');
```

تابع اعمال فیلتر :

```
function output=filtering(image,filter)
    [R,C] = size(image);
    [x,y] = size(filter);
    image = padarray(image,[1,1]);
    output = zeros(R,C,'double');
    for i=1:R
        for j=1:C
            part = image(i:i+x-1,j:j+y-1);
            mult = part.*filter;
            out = sum(mult,'all');
            output(i,j) = out;
        end
    end
end
```

۳.۴.۲

لود کردن تصویر و تعریف فیلتر ها و فراخوانی توابع و نمایش خروجی:

```
image = imread('Images/3/Elaine.bmp');
image = double(image);

filter1 = [1 0;0 -1];
filter2 = [0 1;-1 0];

img1 = filtering(image,filter1);
img2 = filtering(image,filter2);
```

```
imwrite(img1,'out1.png');
imwrite(img2,'out2.png');
```

تابع اعمال فیلتر :

```
function output=filtering(image,filter)
    [R,C] = size(image);
    [x,y] = size(filter);
    image = padarray(image,[1,1]);
    output = zeros(R,C,'double');
    for i=1:R
        for j=1:C
            part = image(i:i+x-1,j:j+y-1);
            mult = part.*filter;
            out = sum(mult,'all');
            output(i,j) = out;
        end
    end
end
```

۳.۵.۱

لود کردن تصویر و تعریف مقادیر ورودی فیلتر:

```
image = imread('Images/3/Elaine.bmp');
filters = [3;5;7;9;11];
alpha = [0.5;1;2];
```

اجرایunsharp masking با مقادیر مختلف و نمایش خروجی :

```
for i=1:size(filters)
    f=filters(i);
    smoothed_img = imgaussfilt(image,f);
    imwrite(smoothed_img,"smoothed_filter"+f+".png");
    for j =1:size(alpha)
        a = alpha(j);
        new_image = image + a*(smoothed_img - image);
        %imwrite(new_image,"alpha"+a+"filter"+f+".png");
        figure
        imshow(new_image);
        title("alpha="+a+",filter="+f+"*"+f);

    end
end
```

پایان :’((((