ECE 5256

Project 5: Advanced Filters

Keefe Kamp

18 February 2023

Figure 1: Input image for the filters before noise.

For this project I looked at advanced filters. The filters I looked at where the arithmetic mean, geometric mean, Midpoint filter, Adaptive filter and Adaptive Medial filter. Each of these filters were applied to a noisy version of the image in Fig. 1. Noise was then added to the input with a standard deviations of 10,20,30, and 40. For each of the amount of noise the filters before we applied and the mean squared error (MSE) was calculated from the original image. At the end the standard deviation vs MSE was plotted. For each other the filters a window of 3x3 was slid across the image and the appropriate filter was used for that window. For the Adaptive Medial filter the window was maxed out at 7x7. The MSE for the difrrent standard deviations of noise behaved interestingly at the lower standard deviation the geometric mean did the worst and the Adaptive filter did the best both my MSE and visually. The geometric and arithemtic means blured the image severly. As the standard devation of the noise incresed the filteres preforemd diffrently. At the highest standard devation the adative filter did the best and the Adaptive medial filter did the worst. In general the MSE increesed with sigma which is expected as there is more noise the filter had to deal with. The trends can be see in Fig. 2.

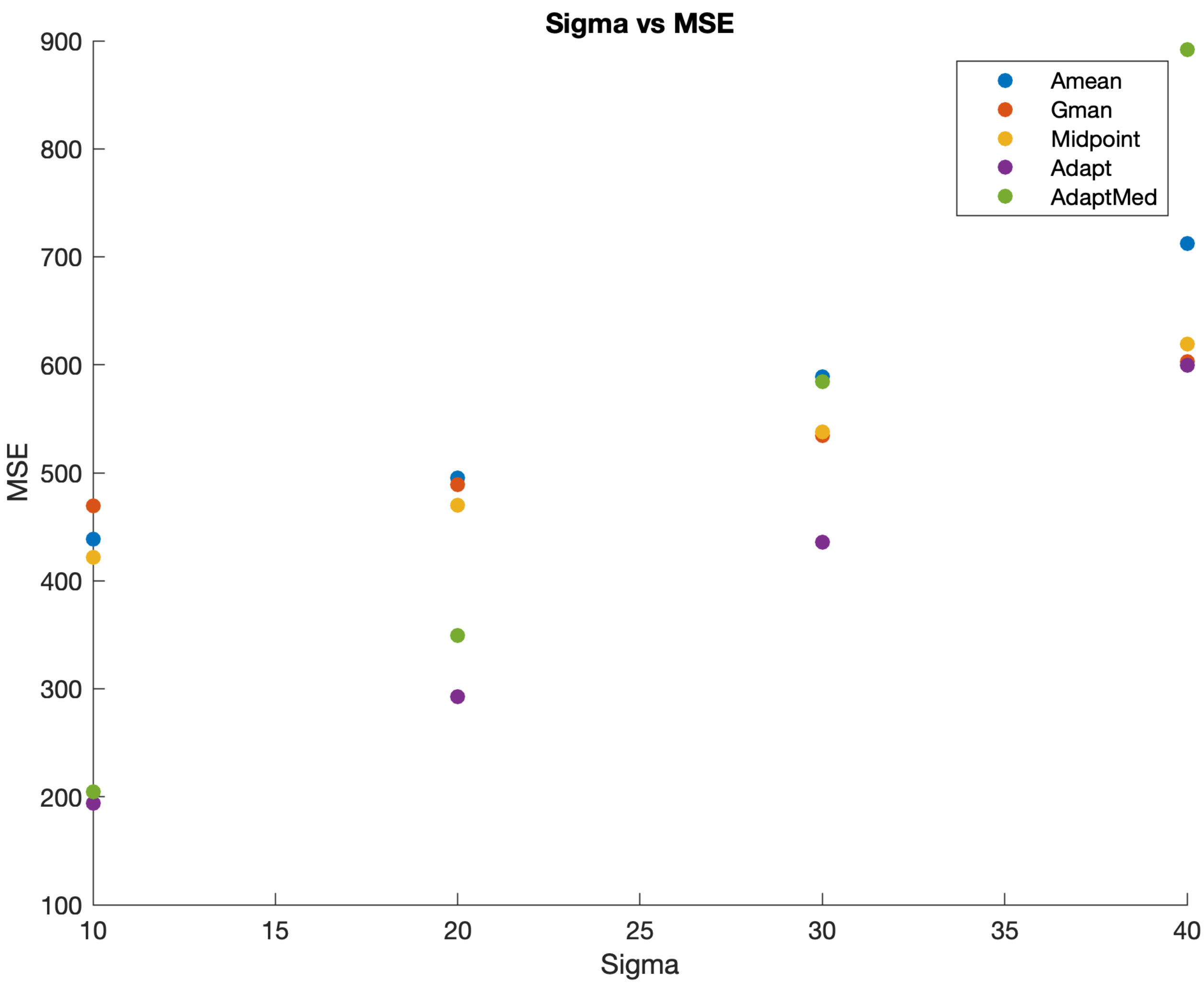

Figure 2: Standard Devation (σ) vs Mean Squared error for each filter.

A MATLAB Code

This is the code used for the coding portions of this project.

```
%Housekeeping commands
clear all
close all
Drops=imread('City.jpeg'); %reading in image
Drops=im2gray(Drops); %converting to grayscale
figure %displaying input image
imagesc(Drops)
axis off image
colormap('gray')
title('Input Image')
exportgraphics(gcf,'Input.png','Resolution',300)
MSEs=zeros(4,5); %allocating memory for MSEs
for l=1:4
    k=10*l; %setting standard dev for noise
    noise=uint8(floor(randn(length(Drops)).*k)); %making noise array
    nvar=var(cast(noise,'double'),0,'all'); %reclacluating var for noise
    NDrops=Drops+noise; %adding noise
    figure %displaying and saving noisy image
    imagesc(NDrops)
    title({'Noisy Image','sigma:'+string(k)})
    axis off image
    colormap('gray')
    exportgraphics(gcf,'Noisy'+string(k)+'.png','Resolution',300)
    [M,N]=size(Drops);
    NewImage=zeros([M,N]); %creating blank array for new images
    %allocating for new outputs
    AMean=NewImage;
    Gmean=NewImage;
    Midpoint=NewImage;
    Adapt=NewImage;
    AdaptMed=NewImage;
    for j=2:M-2 %sliding for y axis
        for i=2:N-2 %sliding for x axis
```

```
        %applying filtes
        AMean(i,j)=mean(NDrops(i-1:i+1,j-1:j+1),'all');
        Gmean(i,j)=geomean(cast(NDrops(i-1:i+1,j-1:j+1),'double'),'all');
        Midpoint(i,j)=median(NDrops(i-1:i+1,j-1:j+1),'all');
        Adapt(i,j)=AdaptFilter(NDrops(i-1:i+1,j-1:j+1),nvar);
        %applying Adaptive Meanfilter with changing window size
        Zout=-99;
        w=0;
        while Zout==-99 %fail value
            try
            Zout=AdaptMedFilter(NDrops(i-(1+w):i+(1+w),j-(1+w):j+(1+w)));
            w=w+1; %incrsing window size
            catch
                %forcing max window size for edges
                Zout=AdaptMedFilter(NDrops(i-(1+w):i+(1+w),j-(1+w):j+(1+w)),f=true);
            end
        end
        AdaptMed(i,j)=Zout;
    end
end
%displaing each new image
figure
MSE=immse(cast(Drops,'double'),AMean);% Calculating MSE
MSEs(1,1)=MSE; %adding to MSE array
imagesc(AMean)
title({'Arithmetic Mean','sigma:'+string(k), 'MSE:'+string(MSE)})
axis off image
colormap('gray')
exportgraphics(gcf,'AMean'+string(k)+'.png','Resolution',300)
figure
image(Gmean)
MSE=immse(cast(Drops,'double'),Gmean);
MSEs(1,2)=MSE;
title({'Geometric Mean','sigma:'+string(k), 'MSE:'+string(MSE)})
axis off image
colormap('gray')
```

```
    exportgraphics(gcf,'GMean'+string(k)+'.png','Resolution',300)
    figure
    MSE=immse(cast(Drops,'double'),Midpoint);
    MSEs(1,3)=MSE;
    image(Midpoint)
    title({'Midpoint Filter','sigma:'+string(k), 'MSE:'+string(MSE)})
    axis off image
    colormap('gray')
    exportgraphics(gcf,'MidPoint'+string(k)+'.png','Resolution',300)
    figure
    MSE=immse(cast(Drops,'double'),Adapt);
    MSEs(1,4)=MSE;
    imagesc(Adapt)
    title({'Adaptive Filter', 'sigma:'+string(k),'MSE:'+string(MSE)})
    axis off image
    colormap('gray')
    exportgraphics(gcf,'Adapt'+string(k)+'.png','Resolution',300)
    figure
    MSE=immse(cast(Drops,'double'),AdaptMed);
    MSEs(1,5)=MSE;
    imagesc(AdaptMed)
    title({'Adaptive Medial Filter','sigma:'+string(k), 'MSE:'+string(MSE)})
    axis off image
    colormap('gray')
    exportgraphics(gcf,'AdaptMed'+string(k)+'.png','Resolution',300)
end
%plotting MSE vs Sigma
figure
c={'Amean','Gman','Midpoint','Adapt','AdaptMed'};
scatter(10:10:40,MSEs,'filled')
legend(c)
xlabel('Sigma')
ylabel('MSE')
title('Sigma vs MSE')
exportgraphics(gcf,'SigmavsMSE.png','Resolution',300)
```

```
function A=AdaptFilter(X,eta) %adaptive filter defination
    ml=mean(X,'all');
    L=var(cast(X,'double'),0,"all");
    g=X(ceil(length(X)/2),ceil(length(X)/2));
    A=g-((eta/L)*(g-ml));
end

function A=AdaptMedFilter(X,f) %Adaptive med filter def
    if nargin < 2 %setting default values
        f = true;
    end
    %calcualting needed values
    zmin=min(X,[],'all');
    zmax=max(X,[],'all');
    zmed=median(X,'all');
    zxy=X(ceil(length(X)/2),ceil(length(X)/2)); %central value
    if zmed==zmin || zmed==zmax
        if length(X)==7
            A=zmed;
        elseif f==true %forceing max window size
            A=zmed;
        else
            A=-99;% for increasing window size
        end
    else
        if zmed==zmin || zmed==zmax
            A=zmed;
        else
            A=zxy;
        end
    end

end
```