THE UNIVERSITY OF TEXAS AT ARLINGTON, TEXAS
DEPARTMENT OF ELECTRICAL ENGINEERING

EE 5356

DIGITAL IMAGE PROCESSING

PROJECT # 12

by

SOUTRIK MAITI

1001569883

Presented to

Dr. K.R.RAO

April 26, 2019

Geometric Mean Filter

MATLAB Code:

```
clc;
clear all;
close all;

%% Reading the image
img =  imread('lena512.bmp');
img = img(:,:,1);

%% Setting the size for the GM filter
Gx_x = size(img);
Nm_0 = Gx_x(1);

%% Plotting the original image
figure(1);
imshow(img);
title('Original Image');
saveas(gca,'Origin.jpg');

%% Converting the image to the Fourier transform

imge_ft = fft2(img);
img3_one = fftshift(imge_ft);
Tx_x = zeros(Nm_0);
Tx_y = zeros(Nm_0);
Tx_z = zeros(Nm_0);

%% Filtering the images with different k

Tx_x = h_fil_ter(0.0025,Nm_0);
Tx_y = h_fil_ter(0.001,Nm_0);
Tx_z = h_fil_ter(0.00025,Nm_0);
Aa_x = randn(Nm_0,Nm_0);
Aax_FT = fft2(Aa_x);
h1_Filtone = fft2(Tx_x);
h2_Filttwo = fft2(Tx_y);
h3_Filtthree = fft2(Tx_z);

Ww_x = img3_one.*Tx_x + Aax_FT;
Img3_Tx = ifft2(ifftshift(Ww_x));
Ww_y = img3_one.*Tx_y + Aax_FT;
Img3_Ty = ifft2(ifftshift(Ww_y));
Ww_z = img3_one.*Tx_z + Aax_FT;
Imge_Tz = ifft2(ifftshift(Ww_z));

%% Displaying the degraded images

figure(2);
```

```
subplot(3,1,1);
imshow(uint8(Img3_Tx));
title('Degraded image with k = 0.0025');

subplot(3,1,2);
imshow(uint8(Img3_Ty));
title('Degraded image with k = 0.001');

subplot(3,1,3);
imshow(uint8(Imge_Tz));
title('Degraded image with k = 0.00025');
saveas(gca,'deg_imgs.jpg');

%% Steps for image restoration

Qq_u = zeros(Nm_0);
Qq_u = abs(fftshift(ifft2(fft2(img).*conj(fft2(img))))./(Nm_0^2);
Qq_v = zeros(Nm_0);
Qq_v = abs(fftshift(ifft2(fft2(Aa_x).*conj(fft2(Aa_x))))./(Nm_0^2);

Qq_w = fftshift(fft2(Qq_u));
Qq_x = fftshift(fft2(Qq_v));

Ee_x = Geo_Mean_filt(Qq_w,Qq_x,Tx_x,0.25,Nm_0);
Ee_y = Geo_Mean_filt(Qq_w,Qq_x,Tx_y,0.25,Nm_0);
Ee_z = Geo_Mean_filt(Qq_w,Qq_x,Tx_z,0.25,Nm_0);

ResultantA = Ww_x.*Ee_x;
ResultantB = Ww_y.*Ee_y;
ResultantC = Ww_z.*Ee_z;

IR_fil_1 = ifft2(ifftshift(ResultantA));
IR_fil_2 = ifft2(ifftshift(ResultantB));
IR_fil_3 = ifft2(ifftshift(ResultantC));

%% Displaying Images with different K values

figure(3);
subplot(3,1,1);
imshow(uint8(IR_fil_1));
title('Restored image with values k = 0.0025 and s= 0.25');

subplot(3,1,2);
imshow(uint8(IR_fil_2));
title('Restored image with values k = 0.001 and s= 0.25');

subplot(3,1,3);
imshow(uint8(IR_fil_3));
title('Restored imagewith values k = 0.00025 and s= 0.25');

saveas(gca,'res_img.jpg');
```

```
% Manually Defined Functions


% Geometric Mean Filter:

function Ff_x = Geo_Mean_filt(aa,bb,cc,dd,ee)

Cc_x = invese_filtr(cc,ee);

Ff_x = ((Cc_x).^dd).*(aa.*conj(cc)./(aa.*(abs(cc).^2)+bb)).^(1-dd);

end



% H Filter:

function Ss_x = h_fil_ter(Ssy,Ssz)
 for Hs_x = 1:Ssz
    for Hs_y = 1:Ssz
        Ss_x(Hs_x,Hs_y) = exp(-Ssy*((Hs_x-Ssz/2)^2+(Hs_y-
Ssz/2)^2)^(5/6));
    end
end

end



% Inverse Filter:

function Cc_x = invese_filtr(Cc_y,Cc_z)
ep = 0.001;
for Sx = 1:Cc_z
    for Sy = 1:Cc_z
        if(Cc_y(Sx,Sy) < ep)
            Cc_x(Sx,Sy) = 0;
        else
            Cc_x(Sx,Sy) = 1/Cc_y(Sx,Sy);
        end
    end
end

end
```

Results:

Original Image

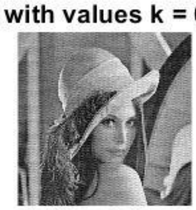
Degraded image with k = 0.0025

Degraded image with k = 0.001

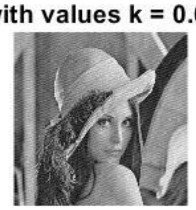
Degraded image with k = 0.00025

Restored image with values k = 0.0025 and s= 0.25

Restored image with values k = 0.001 and s= 0.25



Restored imagewith values k = 0.00025 and s= 0.25



Conclusion:

- The formula for the geometric mean is gicen as:

  $$G = \sqrt[n]{a_1 \cdot a_2 \cdots a_n}$$

- The GMF is better at eliminating Gaussian type noise as well as at the same time conserving edges when compared to the arithmetic mean filter (AMF).
- The GMF is highly prone to negative outliners.
- The GMF is a combination of both the Wiener filter and constrained least squares principles.
- The resultant image obtained at the end of the geometric mean filter is found to be smoother and sharp.
- The filters are easy to develop as they are designed using Fourier domain and the approximate circuit matrix. It is developed using approximate circuit matrix and Fourier Domain. Filters used here are easy to develop, implement and understand.