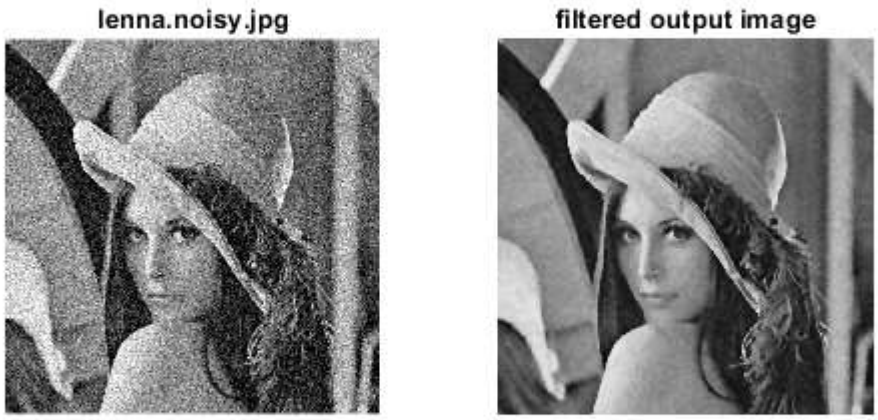
```
clc;
close all;
clear all;
% Reading the input image
I = imread('lenna.noise.jpg');
a = im2gray(I);
[m,n] = size(a);%size of image
f=2;%neighborhood window size = 2f+1 (5*5)
t=3;%similarity window size = 2t +1 (7*7)
% %formation of gausian kernel
sigma=1;          %standard deviation
ker_sum=0;          % sum of element of the kernel
ker_size= 2*f+1;     % size of kernel
% Initiating kernel
ker = zeros(ker_size,ker_size);
for x=1:ker_size
    for y=1:ker_size
        xy = x-f-1;   %horizontal distance
        wz = y-f-1;   % vertical distance
        ker(x,y) = 100*exp(((xy*xy)+(wz*wz))/(-2*(sigma*sigma)));
        ker_sum = ker_sum + ker(x,y);
    end
end
kernel = ker ./ f;
kernel = kernel / ker_sum;   % normalization
% adding noise into image
noisy_img = a;
noisy = double(noisy_img);
%clear output image
out= zeros(m,n);
%Degree of filtering
h=30;
% formation of boundaries for noisy image
noisy1 = padarray(noisy,[f,f],'symmetric');
% calculation of  ouput for each pixel
for i=1:m
    for j=1:n
        im = i+f;   % to compensate for shift due to padarray function
        jn= j+f;
        % neighborhood of prticular pixel
        W1 = noisy1(im-f:im+f , jn-f:jn+f);
        % BOundaries of similarity window
        rmin = max(im-t, f+1);
        rmax = min(im+t, m+f);
        smin = max(jn-t, f+1);
        smax = min(jn+t, n+f);
        % Calculate weighted average next
        NL=0;    % same as cleared
        Z =0;
        %  loop for all the pixels
        for r=rmin:rmax
            for s=smin:smax
                % neighborhood of pixel 'j' being compared for similarity
                W2 = noisy1(r-f:r+f, s-f:s+f);
                % square of weighted euclidian distances
                ed = sum(sum(kernel.*(W1-W2).*(W1-W2)));
                % weight of similarity between both pixels : s(i,j)
                sij = exp(-ed/(h*h));
                % update Z and NL
                Z = Z + sij;
                NL = NL + (sij*noisy1(r,s));
            end
        end
```

```
        % normalization of NL
        out(i,j) = NL/Z;
    end
end
out= uint8(out); %conversion inti uint8
% plotting of image
figure(1);
subplot(1,2,1);
imshow(noisy_img);
title('lenna.noisy.jpg');
subplot(1,2,2);
imshow(out);
title('filtered output image');
```



Published with MATLAB® R2020b