

|  |
| --- |
| **Title:** To implement low pass and high pass filtering in spatial and frequency domain |

**Objective:** To learn and understand the effects of filtering in spatial and frequency domain on images using Matlab.

**Expected Outcome of Experiment:**

|  |  |
| --- | --- |
| **CO** | **Outcome** |
| **CO4** | Design & implement algorithms for digital image enhancement, segmentation & restoration. |

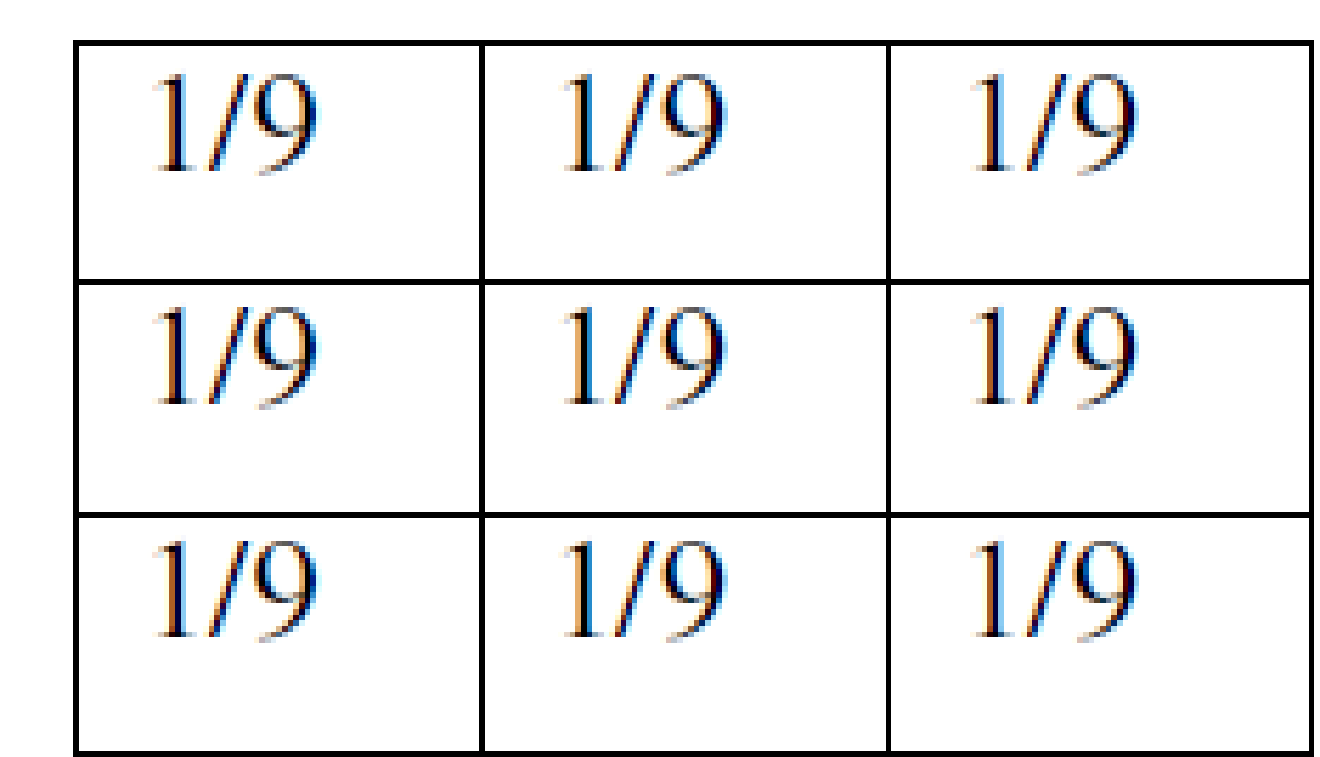
**Books/ Journals/ Websites referred:**

1. http://www.mathworks.com/support/
2. www.math.mtu.edu/~msgocken/intro/intro.html.
3. R. C.Gonsales R.E.Woods, “Digital Image Processing”, Second edition, Pearson Education
4. S.Jayaraman, S Esakkirajan, T Veerakumar “Digital Image Processing “Mc Graw Hill.
5. S.Sridhar,”Digital Image processing”, oxford university press, 1st edition."

**Pre Lab/ Prior Concepts:**

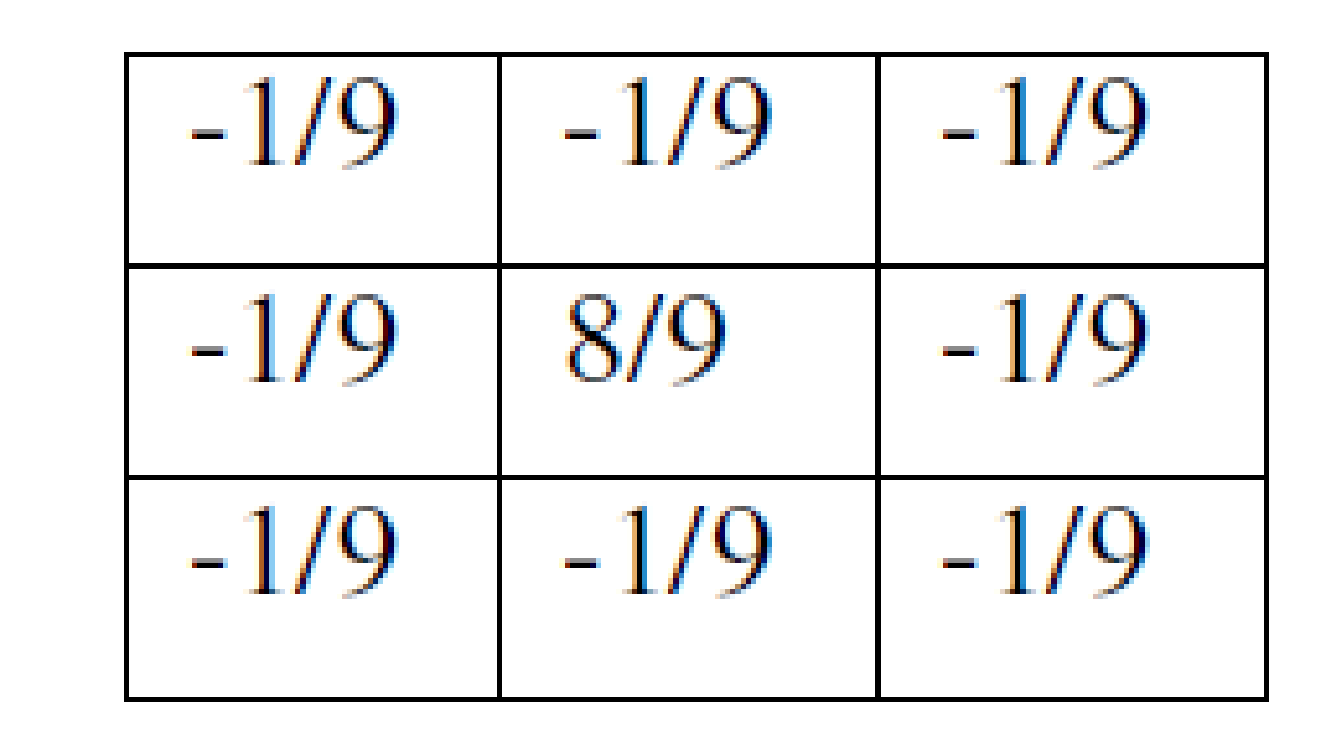
**Filtering in Spatial Domain:**

**Low pass filtering** as the name suggests removes the high frequency content from the image. It is used to remove noise present in the image. Mask for the low pass filter is:



One important thing to note from the spatial response is that all the coefficients are positive. We could also use 5 x 5 or 7 x 7 mask as per our requirement. We place a 3 x 3 mask on the image. We start from the left hand top corner. We cannot work with the borders and hence are normally left as they are. We then multiply each component of the image with the corresponding value of the mask. Add these values to get the response. Replace the centre pixel of the o/p image with these responses. We now shift the mask towards the right till we reach the end of the line and then move it downwards.

**High pass filtering** as the name suggests removes the low frequency content from the image. It is used to highlight fine detail in an image or to enhance detail that has been blurred. Mask for the high pass filter is:



One important thing to note from the spatial response is that sum of all the coefficients is zero. We could also use 5 x 5 or 7 x 7 mask as per our requirement. We place a 3 x 3 mask on the image. We start from the left hand top corner. We cannot work with the borders and hence are normally left as they are. We then multiply each component of the image with the corresponding value of the mask. Add these values to get the response. Replace the centre pixel of the o/p image with these responses. We now shift the mask towards the right till we reach the end of the line and then move it downwards.

**Median filtering** is a signal processing technique developed by tukey that is useful for noise suppression in images. Here the input pixel is replaced by the median of the pixels contained in the window around the pixel. The median filter disregards extreme values and does not allow them to influence the selection of a pixel value which is truly representative of the neighbourhood.

**High boost filtering** is often desirable to emphasize high frequency components representing the image details (by means such as sharpening) without eliminating low frequency components representing the basic form of the signal. In this case, the high-boost filter can be used to enhance high frequency component while still keeping the low frequency components:

\begin{displaymath}
I_{hb}=I_o+c I_{hp}=(W_{ap}+c W_{hp})* I_o=W_{hb} * I_o
\end{displaymath}

where c is a constant and

$W_{hb}=c W_{ap}+W_{hp}$

 is the high boost convolution kernel. For example:

\begin{displaymath}
W_{hb}=W_{ap}+c  W_{hp}
=\left[ \begin{array}{ccc} 0 & 0 ...
... -c & 0  -c & 4c+1 & -c \\
0 & -c & 0 \end{array} \right]
\end{displaymath}

\begin{displaymath}W_{hb}=W_{ap}+c W_{hp}
=\left[ \begin{array}{ccc} 0 & 0 & 0...
... & -c  -c & 8c+1 & -c \\
-c & -c & -c \end{array} \right]
\end{displaymath}

**Implementation Details:**

x = imread('drag.jpg');

subplot(2,4,1);

imshow(x,[]);

title('Original')

x = rgb2gray(x);

x = padarray(x, [1,1], 0, 'both');

noise = x;

subplot(2,4,2);

imshow(x,[]);

title('Grayscale (after padding)')

filtered = noise;

[m,n] = size(noise);

noise2 = imnoise(x, 'gaussian');

subplot(2,4,3);

imshow(noise2,[]);

title('Gaussian noise added');

a = [1, 1, 1; 1, 1, 1; 1, 1, 1];

for i = 2:m-1

for j = 2:n-1

b = [noise2(i-1,j-1)\*a(1,1) noise2(i,j-1)\*a(2,1) noise2(i+1,j-1)\*a(3,1) noise2(i-1,j)\*a(1,2) noise2(i,j)\*a(2,2) noise2(i+1,j)\*a(3,2) noise2(i-1,j+1)\*a(1,3) noise2(i,j+1)\*a(2,3) noise2(i+1,j+1)\*a(3,3)];

filtered(i,j) = sum(b)/9;

end

end

subplot(2,4,4);

imshow(filtered,[]);

title('Low Pass Filter')

filtered1 = noise;

noise = double(noise);

[m,n] = size(noise);

a = [-1, -1, -1; -1, 8, -1; -1, -1, -1];

for i = 2:m-1

for j = 2:n-1

b = [noise(i-1,j-1)\*a(1,1) noise(i,j-1)\*a(2,1) noise(i+1,j-1)\*a(3,1) noise(i-1,j)\*a(1,2) noise(i,j)\*a(2,2) noise(i+1,j)\*a(3,2) noise(i-1,j+1)\*a(1,3) noise(i,j+1)\*a(2,3) noise(i+1,j+1)\*a(3,3)];

filtered1(i,j) = sum(b);

end

end

subplot(2,4,7);

imshow(filtered1,[]);

title('High Pass Filter')

noise1 = imnoise(x, 'salt & pepper');

subplot(2,4,5);

imshow(noise1,[]);

title('Salt & Pepper noise added')

[m,n] = size(noise1);

filtered2 = zeros(m,n);

for i = 2:m-1

for j = 2:n-1

filtered2(i,j) = median([noise1(i-1,j-1) noise1(i,j-1) noise1(i+1,j-1) noise1(i-1,j) noise1(i,j) noise1(i+1,j) noise1(i-1,j+1) noise1(i,j+1) noise1(i+1,j+1)]);

end

end

subplot(2,4,6);

imshow(filtered2,[]);

title('Median Filter')

filtered2 = x;

a = [-1, -1, -1; -1, 9, -1; -1, -1, -1];

a = a/9;

for i = 2:m-1

for j = 2:n-1

b = [noise(i-1,j-1)\*a(1,1) noise(i,j-1)\*a(2,1) noise(i+1,j-1)\*a(3,1) noise(i-1,j)\*a(1,2) noise(i,j)\*a(2,2) noise(i+1,j)\*a(3,2) noise(i-1,j+1)\*a(1,3) noise(i,j+1)\*a(2,3) noise(i+1,j+1)\*a(3,3)];

filtered2(i,j) = sum(b);

end

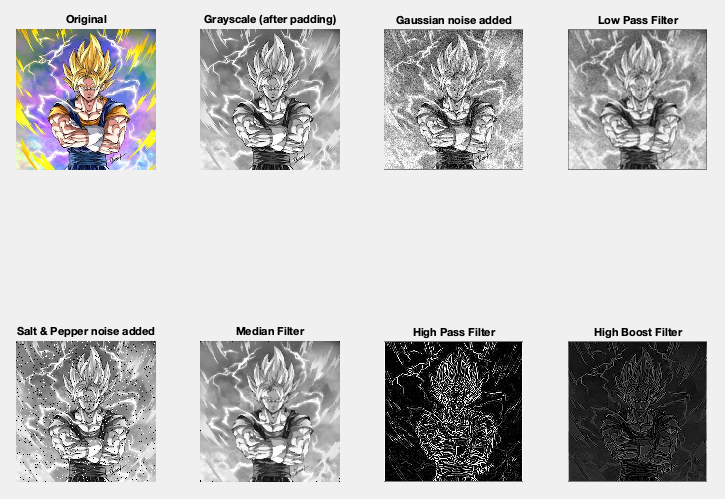
end

subplot(2,4,8);

imshow(filtered2,[]);

title('High Boost Filter')

**Output**

****

**Conclusion:**

Thus, we have understood low pass, high pass, high boost, and median filters and implemented them on an image.

**Post Lab Descriptive Questions**

**1. List & explain different types of noise associated with a digital signal.**

Ans. The various types of noise are as follows-

**Gaussian Noise Model** It is also called as electronic noise because it arises in amplifiers or detectors. Gaussian noise caused by natural sources such as thermal vibration of atoms and discrete nature of radiation of warm objects [5]. Gaussian noise generally disturbs the gray values in digital images. That is why Gaussian noise model essentially designed and characteristics by its PDF or normalizes histogram with respect to gray value

**White Noise Noise** is essentially identified by the noise power. Noise power spectrum is constant in white noise. This noise power is equivalent to power spectral density function. The statement “Gaussian noise is often white noise” is incorrect. However neither Gaussian property implies the white sense. The range of total noise power is -∞ to +∞ available in white noise in frequency domain. That means ideally noise power is infinite in white noise. This fact is fully true because the light emits from the sun has all the frequency components. In white noise, correlation is not possible because of every pixel values are different from their neighbours.

**Electronic noise**

IT staff mainly deals with electronic noise, created in the radio or network systems that transmit data, or in the medium -- such as wire and air -- through which signals are transmitted.

**Thermal noise**

Thermal noise occurs in all transmission media and communication equipment, including passive devices. It arises from random electron motion and is characterized by a uniform distribution of energy over the frequency spectrum with a Gaussian distribution of levels; the higher the temperature of the components or the medium, the greater the level of thermal noise.

**Brownian Noise (Fractal Noise)** Colored noise has many names such as Brownian noise or pink noise or flicker noise or 1/f noise. In Brownian noise, power spectral density is proportional to square of frequency over an octave i.e., its power falls on ¼ th part (6 dB per octave). Brownian noise caused by Brownian motion. Brownian motion seen due to the random movement of suspended particles in fluid.

### Intermodulation noise

Intermodulation (IM) effects result when two or more signals pass through a nonlinear device or medium and interact with each other in ways that produce additional signals, such as harmonics and subharmonics of input signal frequencies.

### Cross-talk

Cross-talk refers to signals interfering with each other electromagnetically. There are essentially three causes of [cross-talk](https://searchnetworking.techtarget.com/definition/crosstalk):

* Electrical coupling between transmission media, like adjacent wires in a multilane serial interface connection -- for example, [Ethernet](https://searchnetworking.techtarget.com/feature/Ethernet-networking-Where-it-is-today-and-whats-next) or [Fibre Channel](https://searchstorage.techtarget.com/definition/Fibre-Channel);
* Poor control of frequency response -- i.e., defective filters or poor filter design;

### Shot noise

Shot noise, also called quantum noise, is the variation in a signal that is caused by the quantized nature of the light and electricity making up the signal. We tend to think of a signal, whether a beam of light or a stream of [electrons](https://whatis.techtarget.com/definition/electron), as being uniform: a steady stream of particles traversing a path.

**Periodic Noise** This noise is generated from electronics interferences, especially in power signal during image acquisition. This noise has special characteristics like spatially dependent and sinusoidal in nature at multiples of specific frequency.

**Impulse Valued Noise (Salt and Pepper Noise)** This is also called data drop noise because statistically its drop the original data values. This noise is also referred as salt and pepper noise. However the image is not fully corrupted by salt and pepper noise instead of some pixel values are changed in the image. Although in noisy image, there is a possibilities of some neighbours does not changed .

**Quantization noise** Quantization noise appearance is inherent in amplitude quantization process. It is generally presents due to analog data converted into digital data. In this noise model, the signal to noise ratio (SNR) is limited by minimum and maximum pixel value, Pmin and Pmax respectively.

**2. Explain with the help of an example how filtering helps in enhancing the quality of an image.**

In image processing filters are mainly used to suppress either the high frequencies in the image, *i.e.* smoothing the image, or the low frequencies, *i.e.* enhancing or detecting edges in the image.

An image can be filtered either in the [frequency](https://homepages.inf.ed.ac.uk/rbf/HIPR2/freqdom.htm) or in the [spatial](https://homepages.inf.ed.ac.uk/rbf/HIPR2/spatdom.htm) domain.

The first involves transforming the image into the frequency domain, multiplying it with the [frequency filter](https://homepages.inf.ed.ac.uk/rbf/HIPR2/freqfilt.htm) function and re-transforming the result into the spatial domain. The filter function is shaped so as to attenuate some frequencies and enhance others. For example, a simple low pass function is *1* for frequencies smaller than the *cut-off frequency* and *0* for all others.

The corresponding process in the [spatial domain](https://homepages.inf.ed.ac.uk/rbf/HIPR2/spatdom.htm) is to [convolve](https://homepages.inf.ed.ac.uk/rbf/HIPR2/convolve.htm) the input image *f*(*i*,*j*) with the filter function *h*(*i*,*j*). This can be written as

Eqn:eqnspin1

The mathematical operation is identical to the multiplication in the frequency space, but the results of the digital implementations vary, since we have to approximate the filter function with a discrete and finite [kernel](https://homepages.inf.ed.ac.uk/rbf/HIPR2/kernel.htm).

The discrete convolution can be defined as a *`shift and multiply'* operation, where we shift the kernel over the image and multiply its value with the corresponding pixel values of the image. For a square [kernel](https://homepages.inf.ed.ac.uk/rbf/HIPR2/kernel.htm) with size *M*× *M*, we can calculate the output image with the following formula:

Eqn:eqnspin2