Sharpening Spatial Filters

Since averaging is analogous to **integration**, it is logical to conclude that sharpening could be accomplished by **spatial differentiation.**

This section deals with various ways of defining and implementing operators for Image sharpening by digital differentiation.

Fundamentally, the strength the response of a **derivative operator** is proportional to the degree of discontinuity of the image at the point at which the operator is applied. Thus, **image differentiation enhances edges** and other discontinuities (such as noise) and deemphasizes areas with slowly varying gray- level values.

Use of first derivatives for Image Sharpening (Non linear) (EDGE enhancement)

About two dimensional high pass spatial filters

An edge is the boundary between two regions with relatively distinct grey level properties. The idea underlying most edge detection techniques is the computation of a local derivative operator.

The magnitude of the first derivative calculated within a neighborhood around the pixel of interest, can be used to detect the presence of an edge in an image.

First derivatives in image processing are implemented using the magnitude of the gradient.

For a function f(x, y), the gradient of f at coordinates (x, y) is defined as the two-dimension

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}.$$

The magnitude of this vector is given by

$$\begin{aligned} \nabla f &= \text{mag}(\nabla \mathbf{f}) \\ &= [G_x^2 + G_y^2]^{1/2} \\ &= \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{1/2}. \end{aligned}$$

The magnitude M (x,y) of this vector, generally referred to simply as the gradient

is $$\nabla f(x,y) = \text{mag}(\nabla f(x,y)) = \left[\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2\right]^{1/2}$$

Size of M(x,y) is same size as the original image. It is common practice to refer to this image as **gradient image** or simply as gradient.

Common practice is to approximate the gradient with absolute values which is simpler to implement as follows.

$$\nabla f(x,y) \cong \left|\frac{\partial f(x,y)}{\partial x}\right| + \left|\frac{\partial f(x,y)}{\partial y}\right|$$

A basic definition of the **first-order derivative of a one-dimensional function** f(x) is the difference

$$\frac{\delta f}{\delta x} = f(x+1) - f(x).$$

Similarly, we define a second-order derivative as the difference

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x).$$

$$\frac{\partial^2 f}{\partial^2 x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$$\frac{\partial^2 f}{\partial^2 y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

In x and y directions

Sharpening Spatial Filters

First derivative

(1) must be zero in flat segments (areas of constant gray-level values);

(2) must be nonzero at the onset of a gray-level step or ramp; and

(3) must be nonzero along ramps.

Similarly, any definition of a second derivative

(1) must be zero in flat areas;

(2) must be nonzero at the onset and end of a gray-level step or ramp;

(3) must be zero along ramps of constant slope.

The digital implementation of the two-dimensional Laplacian is obtained by summing these two components:

$$\frac{\partial^2 f}{\partial^2 x^2} + \frac{\partial^2 f}{\partial^2 y^2} = \nabla^2 f = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y).$$

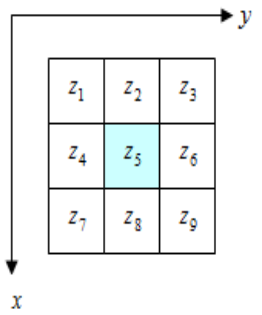
Laplacian operator (for enhancing fine details)

The **Laplacian** of a 2-D function $f(x,y)$ is a second order derivative defined as

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$$

In practice it can be also implemented using a 3x3 mask $\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8)$

Consider a pixel of interest $f(x,y) = z_5$ and a rectangular neighborhood of size $3 \times 3 = 9$ pixels (including the pixel of interest) as shown below.



The main disadvantage of the Laplacian operator is that it produces double edges

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

(a) Filter mask used to implement the digital Laplacian

0	−1	0
−1	4	−1
0	−1	0

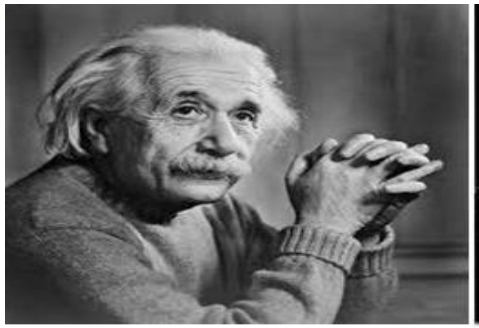
−1	−1	−1
−1	8	−1
−1	−1	−1

$$\nabla^2 f = \left[f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)\right] - 4f(x,y).$$

(b) Mask used to implement an extension that includes the diagonal neighbors.

(d) Two other implementations of the

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is negative} \\ f(x,y) + \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is positive.} \end{cases}$$



LAPLACIAN + ADDITION WITH ORIGINAL IMAGE DIRECTLY

0	-1	0
-1	5	-1
0	-1	0

=

0	0	0
0	1	0
0	0	0

+

0	-1	0
-1	4	-1
0	-1	0

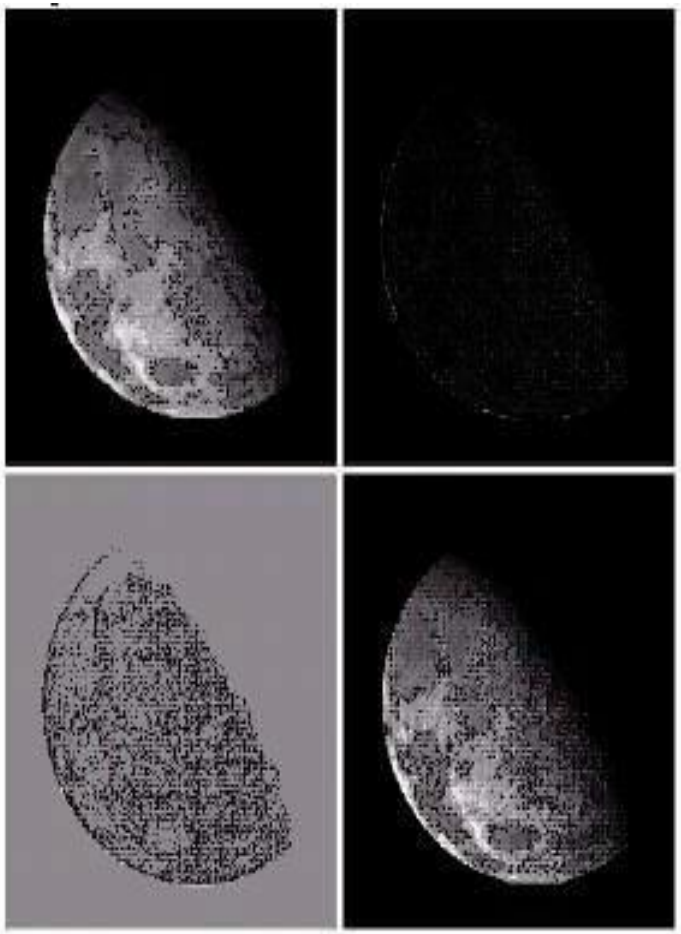
-1	-1	-1
-1	9	-1
-1	-1	-1

=

0	0	0
0	1	0
0	0	0

+

-1	-1	-1
-1	8	-1
-1	-1	-1



- a). image of the North pole of the moon
- b). Laplacian-filtered image with

1	1	1
1	-8	1
1	1	1

- c). Laplacian image scaled for display purposes
- d). image enhanced by addition with original image

Use of first derivatives for Image Sharpening (Non linear)

About two dimensional high pass spatial filters

An edge is the boundary between two regions with relatively distinct grey level properties. The idea underlying most edge detection techniques is the computation of a local derivative operator.

The magnitude of the first derivative calculated within a neighborhood around the pixel of interest, can be used to detect the presence of an edge in an image.

First derivatives in image processing are implemented using the magnitude of the gradient.

For a function f(x, y), the gradient of *f* at coordinates (x, y) is defined as the two-dimensional column vector

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}.$$

The magnitude of this vector is given by

$$\begin{aligned} \nabla f &= \text{mag}(\nabla \mathbf{f}) \\ &= [G_x^2 + G_y^2]^{1/2} \\ &= \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{1/2}. \end{aligned}$$

The magnitude M (x,y) of this vector, generally referred to simply as the gradient ∇f is

$$\nabla f(x,y) = \text{mag}(\nabla f(x,y)) = \left[\left(\frac{\partial f(x,y)}{\partial x} \right)^2 + \left(\frac{\partial f(x,y)}{\partial y} \right)^2 \right]^{1/2}$$

Size of M(x,y) is same size as the original image. It is common practice to refer to this image as **gradient image** or simply as gradient.

Common practice is to approximate the gradient with absolute values which is simpler to implement as follows.

$$\nabla f(x,y) \cong \left| \frac{\partial f(x,y)}{\partial x} \right| + \left| \frac{\partial f(x,y)}{\partial y} \right|$$

Gradient Mask

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

- simplest approximation, 2x2

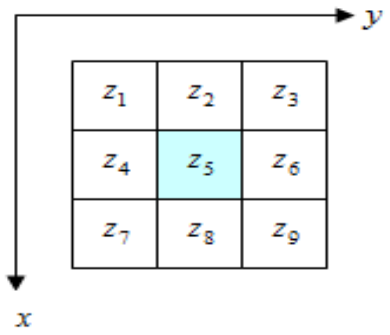
$$G_x = (z_8 - z_5) \quad \text{and} \quad G_y = (z_6 - z_5)$$

$$\nabla f = [G_x^2 + G_y^2]^{1/2} = [(z_8 - z_5)^2 + (z_6 - z_5)^2]^{1/2}$$

$$\nabla f \approx |z_8 - z_5| + |z_6 - z_5|$$

DERIVATIVE OPERATORS

Consider a pixel of interest $f(x,y)=z_5$ and a rectangular neighborhood of size $3\times3=9$ pixels (including the pixel of interest) as shown below.



Roberts operator

Above Equation can be approximated at point Z_5 in a number of ways. The simplest is to use the difference $(Z_5 - Z_8)$ in the x direction and $(Z_5 - Z_6)$ in the y direction. This approximation is known as the **Roberts** operator, and is expressed mathematically as follows

$$\nabla f \cong |z_5 - z_8| + |z_5 - z_6|$$

Another approach for approximating the equation is to use cross differences

$$\nabla f \cong |z_5 - z_9| + |z_6 - z_8|$$

Above Equations can be implemented by using the following masks. The original image is convolved with both masks separately and the absolute values of the two outputs of the convolutions are added.

1	0
-1	0

1	-1
0	0

Roberts operator

1	0
0	-1

0	1
-1	0

Roberts operator

- Roberts cross-gradient operators, 2x2

$$G_x = (z_9 - z_5) \quad \text{and} \quad G_y = (z_8 - z_6)$$

$$\nabla f = [G_x^2 + G_y^2]^{1/2} = [(z_9 - z_5)^2 + (z_8 - z_6)^2]^{1/2}$$

$$\nabla f \approx |z_9 - z_5| + |z_8 - z_6|$$

-1	0
0	1

0	-1
1	0

Prewitt operator

$$\nabla f(x, y) \cong \left|\frac{\partial f(x, y)}{\partial x}\right| + \left|\frac{\partial f(x, y)}{\partial y}\right|$$

Another approximation to the above equation, but using a 3 x 3 matrix is:

$$\nabla f \approx \left|(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)\right| + \left|(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)\right|$$

The difference between the first and third rows approximates the derivative in the *x direction*

- The difference between the first and third columns approximates the derivative in the *y direction*
- The *Prewitt operator masks may be used to implement the* above approximation

-1	-1	-1
0	0	0
1	1	1

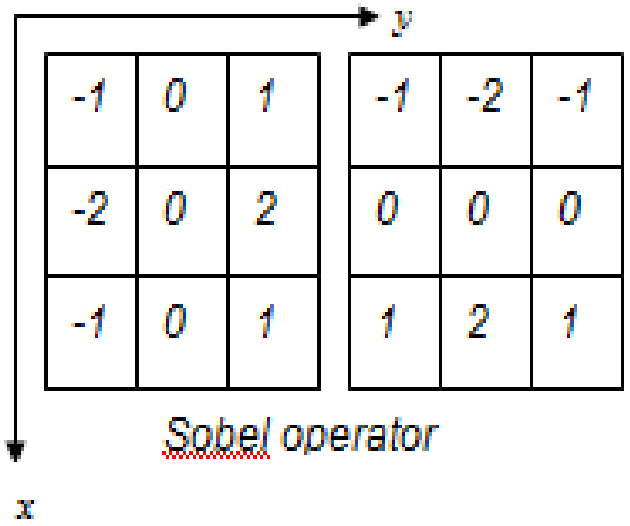
-1	0	1
-1	0	1
-1	0	1

Sobel operator.

Definition and comparison with the Prewitt operator (gives weightage to centre pixel)
The most popular approximation of equation (1) but using a 3×3 mask is the following.

$$\nabla f \cong |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|$$

This approximation is known as the Sobel operator.



-1	0	1
-2	0	2
-1	0	1

-1	-2	-1
0	0	0
1	2	1

Sobel operator

If we consider the left mask of the Sobel operator, this causes differentiation along the y direction.

- Sobel operators, 3x3

$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

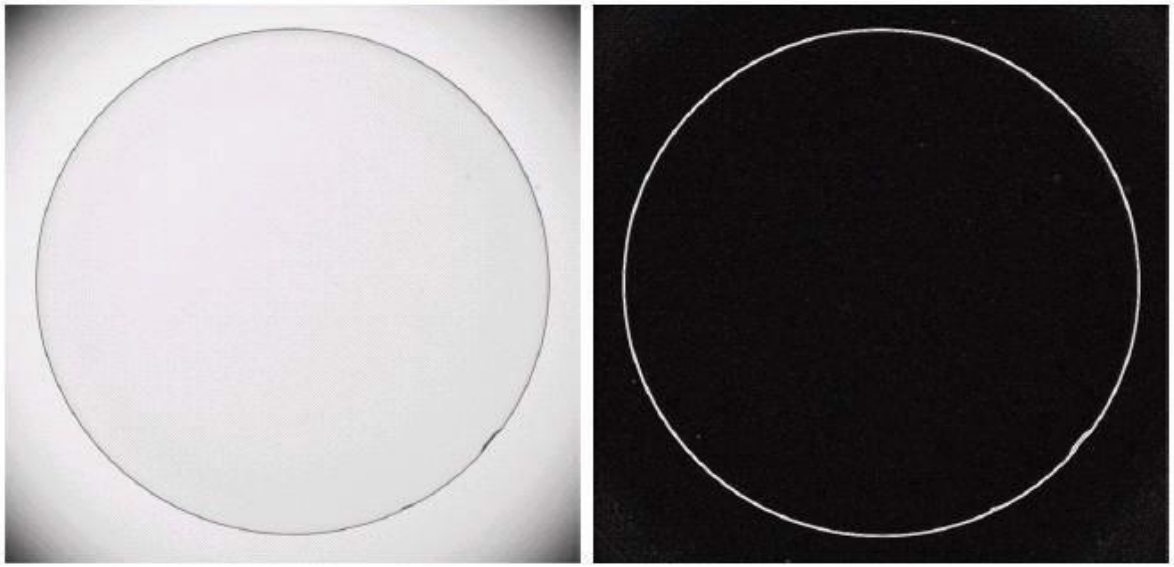
$$\nabla f \approx |G_x| + |G_y|$$

the weight value 2 is to achieve smoothing by giving more important to the center point

-1	-2	-1	-1	0	1
0	0	0	-2	0	2
1	2	1	-1	0	1

150

the summation of coefficients in all masks equals 0, indicating that they would give a response of 0 in an area of constant gray level



a b

FIGURE 3.45
Optical image of contact lens (note defects on the boundary at 4 and 5 o'clock). (b) Sobel gradient. (Original image courtesy of Mr. Pete Sites, Perceptics Corporation.)

Filtering in the Frequency Domain

Filters in the frequency domain can be divided in four groups: Low pass filters

…….IMAGE BLUR

Remove frequencies away from the origin
Commonly, frequency response of these filters is symmetric around the origin;
The largest amount of energy is concentrated on low frequencies, but it represents just image luminance and visually not so important part of image.

High pass filters ……EDGES DETECTION
Remove signal components around and further away from origin
Small energy on high frequency corresponds to visually very important image features such as edges and details. Sharpening = boosting high frequency pixels

Band pass filters
Allows frequency in the band between lowest and the highest frequencies;

Stop band filters

Remove frequency band.

To remove certain frequencies, set their corresponding F(u) coefficients to zero

Low pass filters (smoothing filters)

Ideal Low Pass filters ILPF
Butterworth low pass filters BLPF
Gaussian low pass filters GLPF

High Pass Filters (Sharpening filters)

Ideal High pass filters IHPF
Butterworth High pass filters BHPF
Gaussian High pass filters Laplacian in frequency domain GHPF
High boost , high frequency emphasis filters

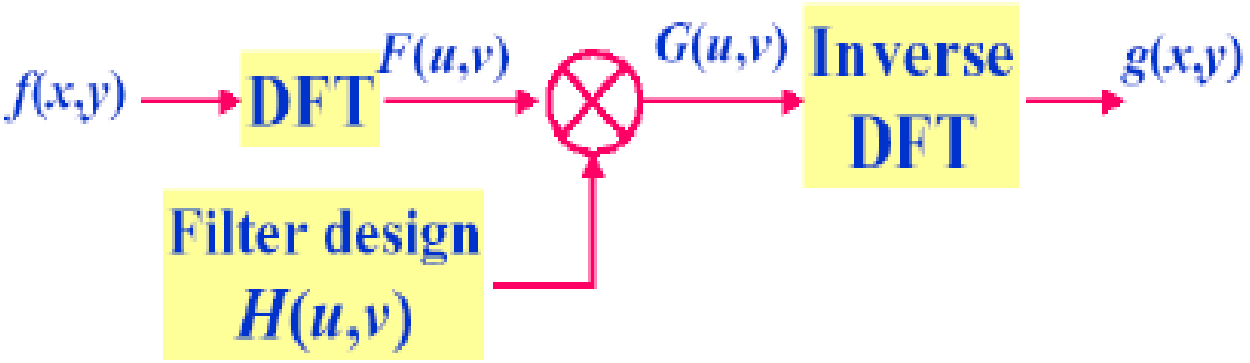
Homomorthic filters

log n or ln

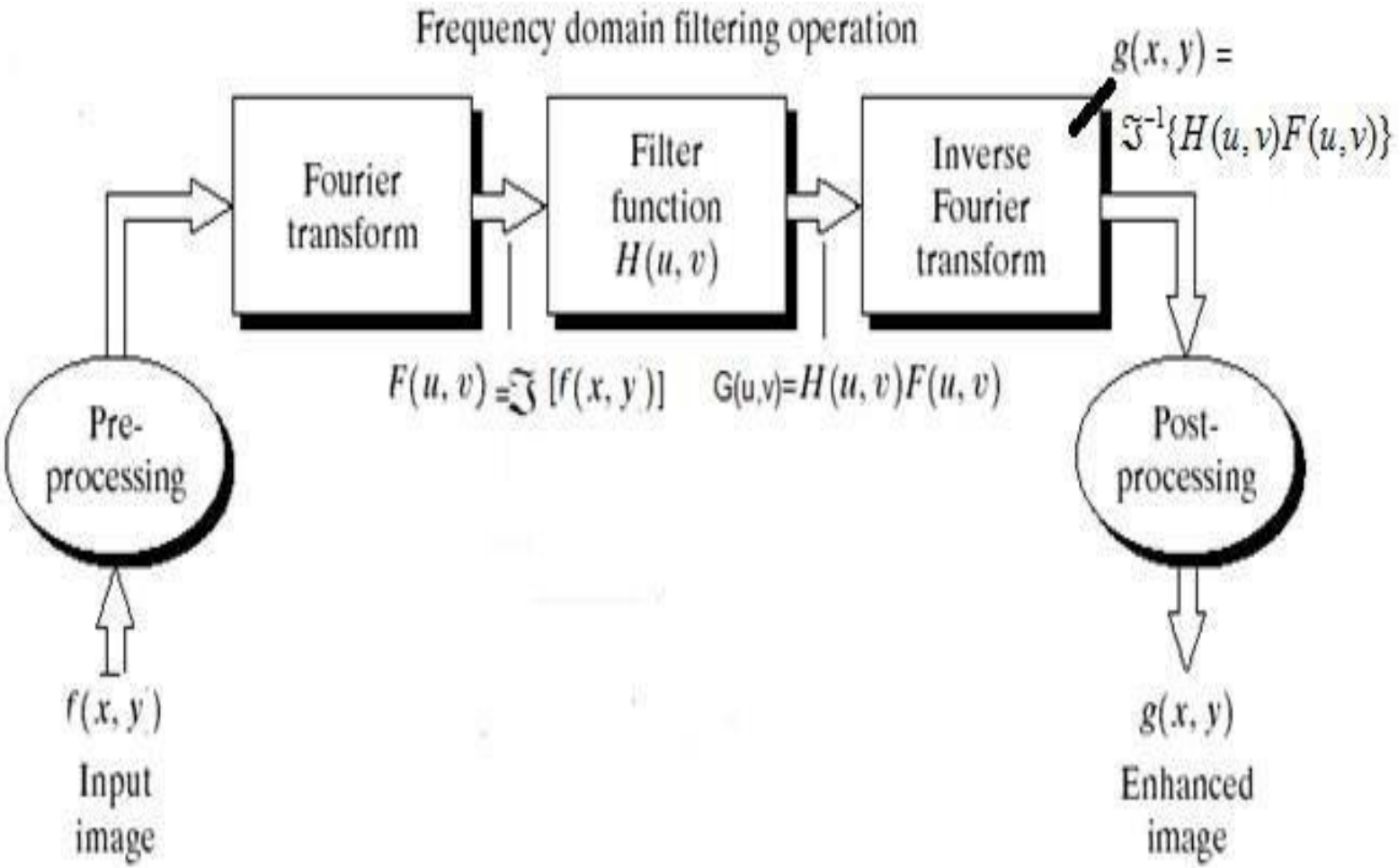
f(x,y)= i(x,y) r(x,y)

F[f(u,v)]= F [log n [i(x,y) r(x,y)] = F [log n [i(x,y)] + F [log n [r(x,y)]

Basic steps for filtering in the frequency domain

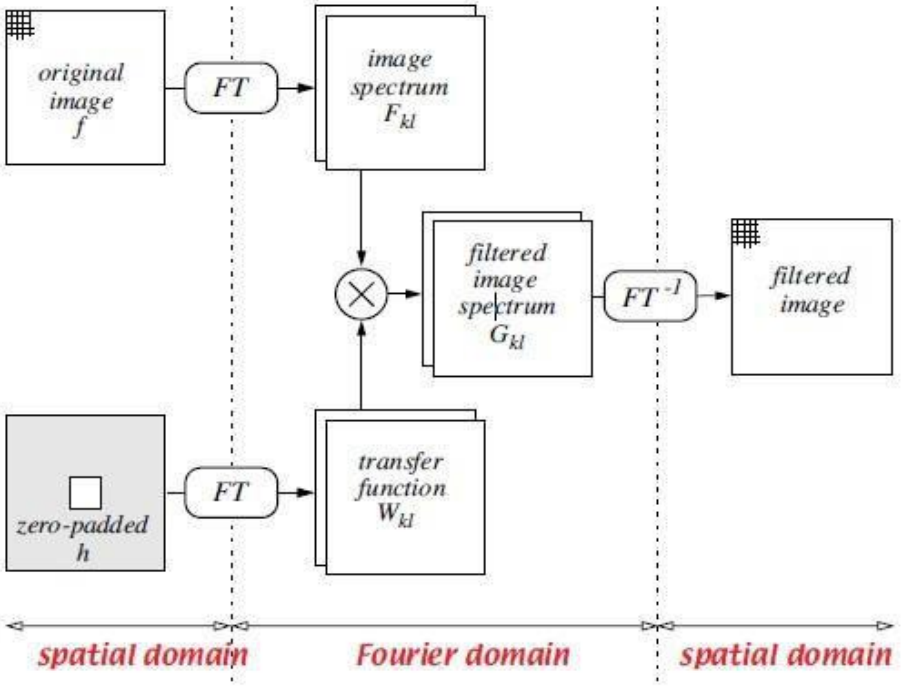


- $H(u,v)$ plays an important role⇨ determine how the enhanced image looks like
- A lowpass $H(u,v)$ results in a *blurring* effect; a highpass $H(u,v)$ results in a *sharpening* effect
- High frequency: edges and sharp details in an image
- Low frequency: slowly varying characteristics of an image (e.g. overall contrast and average intensity)



Basic steps for filtering in the frequency domain.

IMAGE ENHANCEMENT III (Fourier)



Note, multiplication in Fourier domain is full complex operation

Filtering in the Frequency Domain

•***Basic Steps for zero padding***

Zero Pad the input image f(x,y) to p =2M-1, and q=2N-1, if arrays are of same size.

If functions f(x,y) and h(x,y) are of size MXN and KXL, respectively, choose to pad with zeros:

P ш M + N - 1

Q ш K + L- 1

Zero-pad h and f

- Pad both to at least
- Radix-2 FFT requires power of 2

For example, if M = N = 512 and K = L = 16, then P = Q = 1024

- Results in linear convolution
- Extract center MxN

Practical implementation: Overlap-add partitions image into I x J smaller blocks, pads each block and filter h to same size, filters each block separately, and recombines:

Filtering in the Frequency Domain

•***Basic Steps for Filtering in the Frequency Domain:***

1. Multiply the input padded image by (-1)$^{x+y}$ to center the transform.
2. Compute F(u,v), the DFT of the image from (1).
3. Multiply F(u,v) by a filter function H(u,v).
4. Compute the inverse DFT of the result in (3).
5. Obtain the real part of the result in (4).
6. Multiply the result in (5) by (-1)$^{x+y}$.

Given the filter H(u,v) (filter transfer function OR filter or filter function) in the frequency domain, the Fourier transform of the output image (filtered image) is given by:

G (u,v)= H (u,v) F (u,v) ***Step (3) is array multiplication***

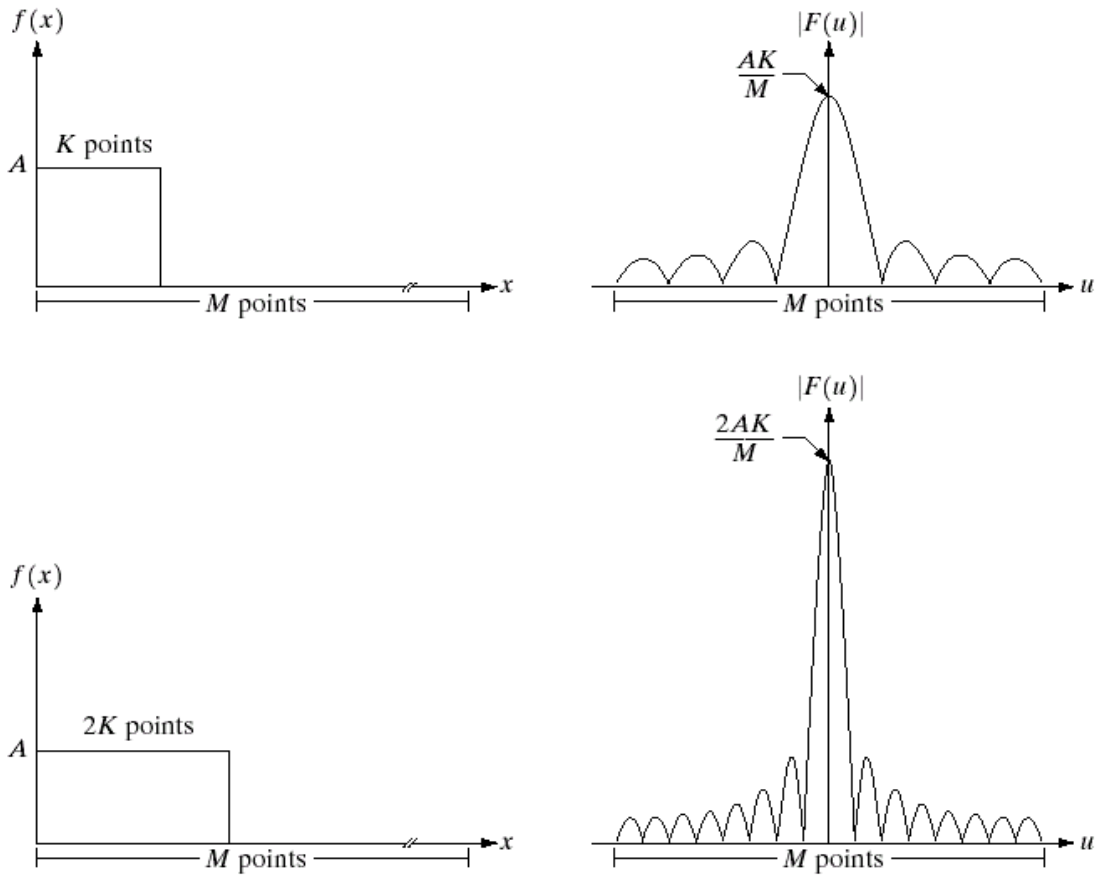
The filtered image g(x,y) is simply the inverse Fourier transform of G(u,v).

g (x,y) = F^{-1} [G(u,v)] = F^{-1} [H (u,v) F (u,v)] ***Step (4)***

$$g_p(x, y) = \{\text{real}\left[\Im^{-1}[G(u, v)]\right]\}(-1)^{x+y}$$

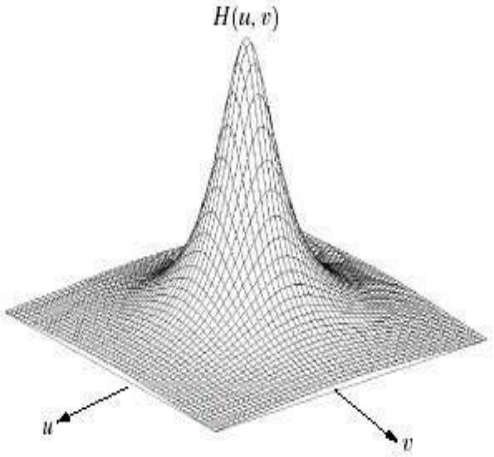
F, H, g, are arrays of same size as in input image. F^{-1} is IDFT.

1. Multiply the input image by $(-1)^{x+y}$ to center the transform
2. Compute F(u,v), the DFT of the image from (1)
3. Multiply F(u,v) by a filter function H(u,v)
4. Compute the inverse DFT of the result in (3)
5. Obtain the real part of the result in (4)
6. Multiply the result in (5) by $(-1)^{x+y}$

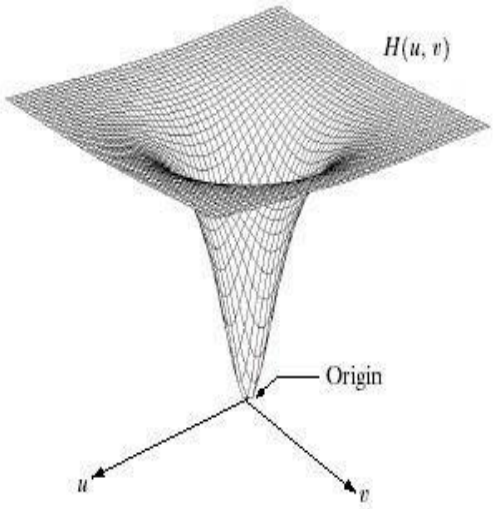


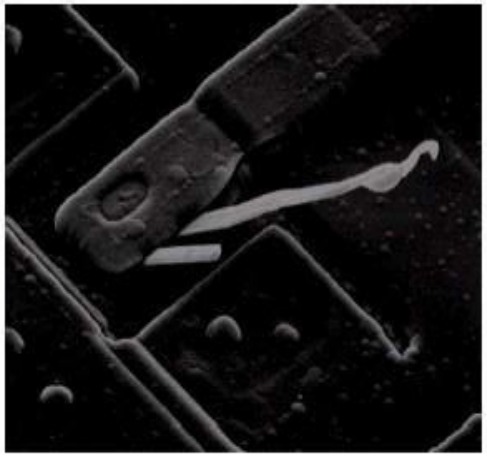
a b
c d

FIGURE 4.2 (a) A discrete function of M points, and (b) its Fourier spectrum. (c) A discrete function with twice the number of nonzero points, and (d) its Fourier spectrum.



Low Pass Filter attenuate high frequencies while passing low frequencies.





High Pass Filter attenuate low frequencies while passing high frequencies.

a b
c d

(a) A two-dimensional lowpass filter function. (b) Result of lowpass filtering the image
(c) A two-dimensional highpass filter function. (d) Result of highpass filtering the image

Correspondence between filtering in spatial and frequency domains

Filtering in frequency domain is multiplication of filter times fourier transform of the input image

G (u,v)= H (u,v) F (u,v)

Let us find out equivalent of frequency domain filter H (u,v) in spatial domain.

Consider f (x,y) = ω(x,y), we know f(u,v) =1
Then filtered output F^{-1} [H (u,v) F (u,v)] = F^{-1} [H (u,v)]

But this is inverse transform of the frequency domain filter
But this is nothing but filter in the spatial domain.
Conversely, if we take a forward Fourier transform of a spatial filter, we get its fourier domain representation

Therefore, two filters form a transform pair
h(x,y) ⟺ H (u,v),

Since h(x,y) can be obtained from the response of a frequency domain filter to an impulse, h(x,y) spatial filter is some times referred as r finite impulse response filter (FIR) of H(u,v)

$f(x,y) * h(x,y) \Leftrightarrow F(u,v)\,H(u,v)$

Spatial domain processing, in general, is faster than frequency domain processing.

In some cases, it is desirable to generate spatial domain mask that approximates a given frequency domain filter.

The following procedure is one way to create these masks in a *least square error sense.*

Recall that filter processing in frequency domain, which is product of filter and function, becomes convolution of function and filter in spatial domain.

$$G(u,v) = H(u,v)F(u,v)$$

Output image — *Filter Transfer Function* — *Input image*

➢ The analogous operation in spatial domain can be implemented as follow

$$g(x,y) = \sum_{i=0}^{N-1}\sum_{k=0}^{N-1} h(x-i, y-k)\, f(i,k)$$

with x, y = 0, 1, 2, ……., *N*-1.

➢ Note that all functions are properly extended to generate a linear convolution in frequency domain.

➢ h is the spatial domain representation of the $H(u,v)$ and it is called *spatial convolution mask.*

➢ The relationship between $h(x,y)$ and $H(u,v)$ is defined as follow

$$H(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} h(x,y)\, e^{-j2\pi\frac{ux+vy}{N}}$$

where $u, v = 0, 1, 2, \ldots\ldots, N-1$.

➢ Now, if we restrict $h(x,y)$ as follow

$$\hat{h}(x,y) = \begin{cases} h(x,y), & 0 \le x, y \le n-1 \\ 0 \quad , & n \le x, y \le N-1 \end{cases}$$

➢ This restriction in effect creates an n x n convolution mask $\hat{h}$, with Fourier transform of

$$\hat{H}(u,v) = \frac{1}{N}\sum_{x=0}^{n-1}\sum_{y=0}^{n-1}\hat{h}(x,y)\,e^{-j2\pi\frac{ux+vy}{N}}$$

where $u, v = 0, 1, 2, \ldots\ldots, N\text{-}1$.

➢ The objective is to find the coefficients of $\hat{h}(x,y)$ such that mean square error is minimized.

$$e^2 = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\left|\hat{H}(u,v) - H(u,v)\right|^2$$

Consider the following filter transfer function:

$$H(u,v) = \begin{cases} 0 & if\ (u,v) = (M/2, N/2) \\ 1 & otherwise \end{cases}$$

This filter will set F(0,0) to zero and leave all the other frequency components. Such a filter is called the notch filter, since it is constant function with a hole (notch) at the origin.

HOMOMORPHIC FILTERING

an image can be modeled mathematically in terms of illumination and reflectance as follow:

$f(x,y) = I(x,y)\, r(x,y)$

Note that:

$F\{ f(x, y)\} \neq F\{i(x, y)\}\ F\{r(x, y)\}$

To accomplish separability, first map the model to natural log domain and then take the Fourier transform of it. $z(x, y) = ln\{ f(x, y)\} = ln\{i(x, y)\} + ln\{r(x, y)\}$

Then,

$F\{z(x, y)\} = F\{ln\ i(x, y)\} + F\{ln\ r(x, y)\}$

or

$Z(u, v) = I(u, v) + R(u, v)$

Now, if we process Z(u,v) by means of a filter function H(u,v) then,

➢ Now, if we process Z(u,v) by means of a filter function H(u,v) then,

$$S(u,v) = H(u,v)Z(u,v) =$$
$$= H(u,v)I(u,v) + H(u,v)R(u,v)$$

➢ Taking inverse Fourier transform of S(u,v) brings the result back into natural log domain,

$$s(x,y) = F^{-1}\{S(u,v)\}$$
$$= F^{-1}\{H(u,v)I(u,v)\} + F^{-1}\{H(u,v)R(u,v)\}$$

By letting

$$i'(x,y) = F^{-1}\{H(u,v)I(u,v)\}$$
$$r'(x,y) = F^{-1}\{H(u,v)R(u,v)\}$$

- Now, to get back to spatial domain, we need to get inverse transform of natural log, which is exponential,

$$s(x,y) = i'(x,y) + r'(x,y)$$
$$g(x,y) = \exp[s(x,y)]$$
$$= \exp[i'(x,y)] \cdot \exp[r'(x,y)]$$
$$= i_o(x,y) r_o(x,y)$$

Where $i_o(x,y)$ is illumination and $r_o(x,y)$ is reflectance components of the output image.

- This method is based on a special case of a class of systems known as *homomorphic systems*.

- The overall model in block diagram will look as follow: