7

Spatial and Intensity Resolution Implementation in MATLAB

❖ Spatial and Gray-level (Intensity) Resolution:

- Spatial resolution is the smallest discernable detail in an image.
- Gray-level resolution is the smallest discernable change in gray(intensity) level.
- For spatial resolution to say a image has a 512x 512 pixels resolution is not a meaningful statement without stating the spatial dimensions encompassed by the image.

❖ Spatial/Coordinate Resolution:

Original image 512*512

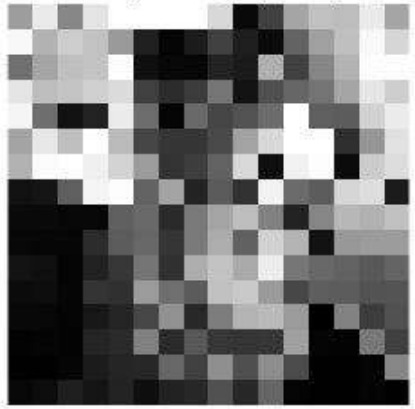
Sample image (256*256)

Sample image (128*128)

Sample image (64*64)

Sample image (32*32)

Sample image (16*16)



A 512 x 512, 8 bit image subsampled down to size 256 x 256, 128 x 128, 64 x 64 , 32 x 32 and 16 x 16 pixels. The number of intensity levels kept at 256.

❖ Aliasing (Down Sampling) :

- Shannon sampling theorem tells us that, if the function is sampled at a rate equal to or greater than twice its highest frequency ($f_s \geq 2f_m$)it is possible to recover completely the original function from the samples.

- If the function is downsampled, then a phenomenon called aliasing(distortion if two pattern or spectrum overlap, the overlapped portion is called aliased) corrupts the sampled image.

- The corruption is in the form of additional frequency components being introduced into the sampled function. These are called aliased frequencies. Downsampling results in a “Checker board” pattern due to reduction of samples.

❖ Up Sampling :

- For zooming an image we need to do interpolation. Interpolation is the process of using know data to estimate values of unknown locations.

(1) Nearest neighbor interpolation:

Assigns the value of closest pixel to the new pixel.

(2) Bilinear interpolation:

Bilinear interpolation considers the closest 2x2 neighborhood of known pixel values surrounding the unknown pixel. It then takes a weighted average of these 4 pixels to arrive at its final interpolated value.

(3) Bicubic interpolation:

It considers the closest 4 x 4 neighborhood of known pixels and takes the weighted average of these 16 pixels.

Original Image (64x64)

Zoomed Image (512x512), using Nearest neighbor interpolation

Zoomed Image (512x512), using Bilinear interpolation

Zoomed Image (512x512), using Bicubic interpolation

❖ Gray-Level/Intensity Resolution :

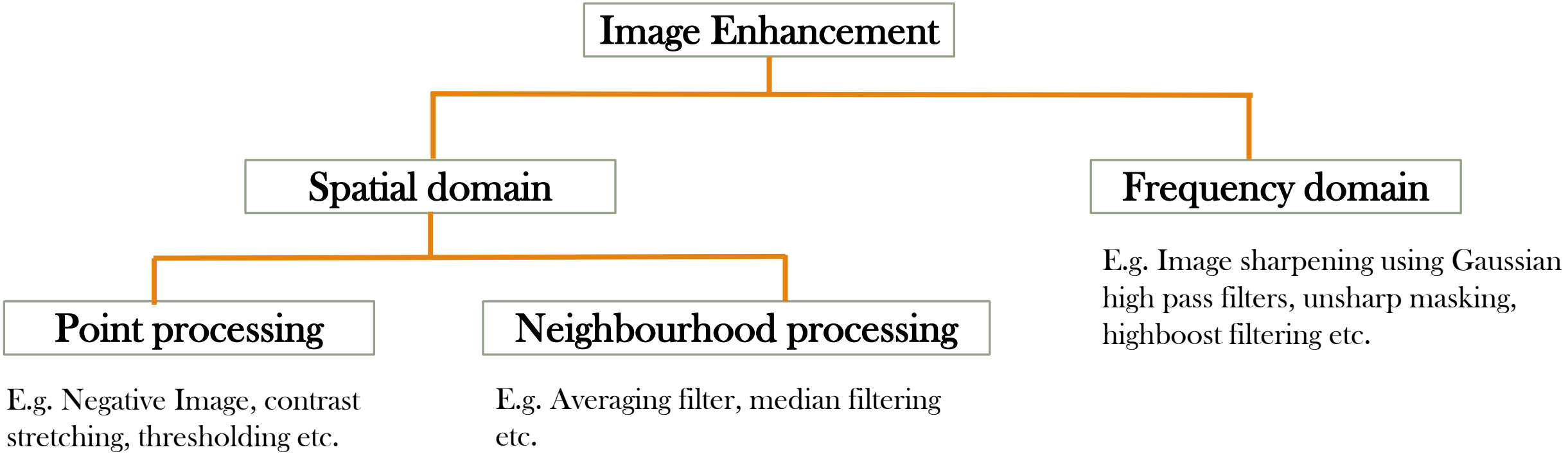
Fig: 512 x 512, 256 level(8 bit) image. Further displayed in 128 level (7 bit), 64 level (6 bit), 32 level (5 bit), 16 level (4 bit), 8 level (3 bit), 4 level (2 bit) and 2 level (1 bit), while keeping spatial resolution constant.

Thank You

8

The basics of Intensity transformations and spatial filtering and implementation in MATLAB

- Image enhancement is the process of enhancing image so that the resultant image is more suitable than original for specific applications.



- The spatial domain processes can be denoted by the expression, g(x, y) = T[f(x, y)], where f(x, y) is the input image, g(x, y) is the output image, and T is an operator on f defined over a neighborhood of point (x, y).

- The smallest possible neighborhood is of size 1 x 1. Here, g depends only on the value of f at a single point (x, y) and T becomes s = T(r), where, for simplicity in notation, s and r are variables denoting, respectively, the intensity of g and f at any point (x, y).

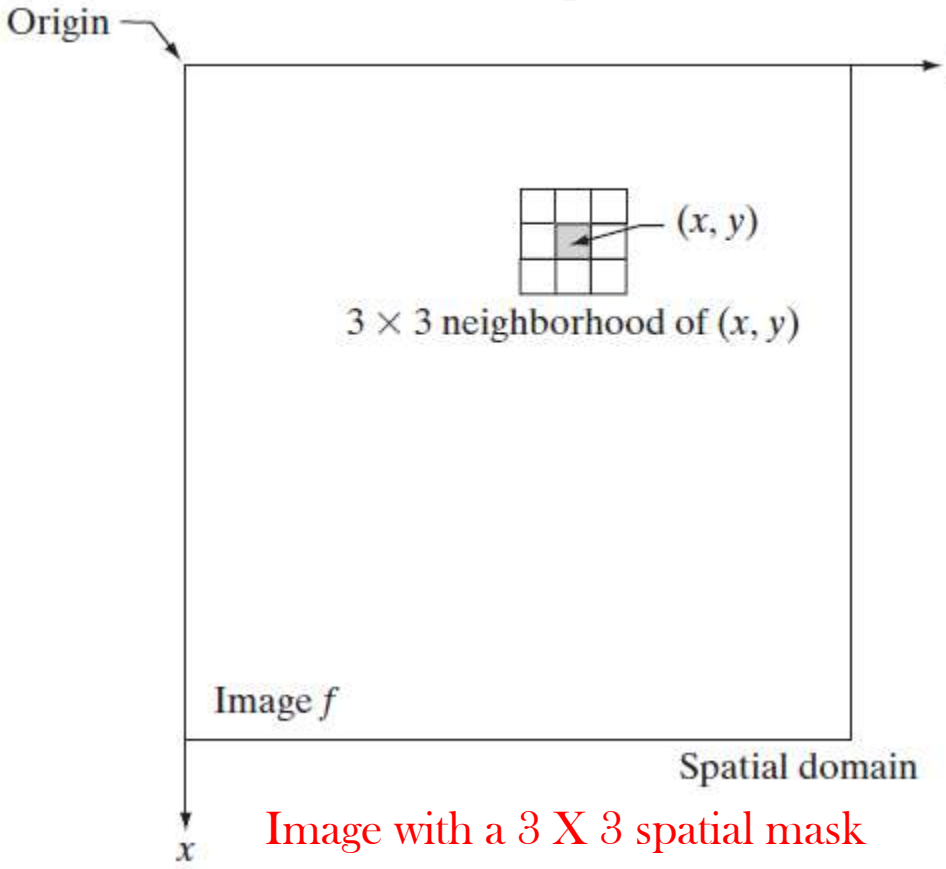
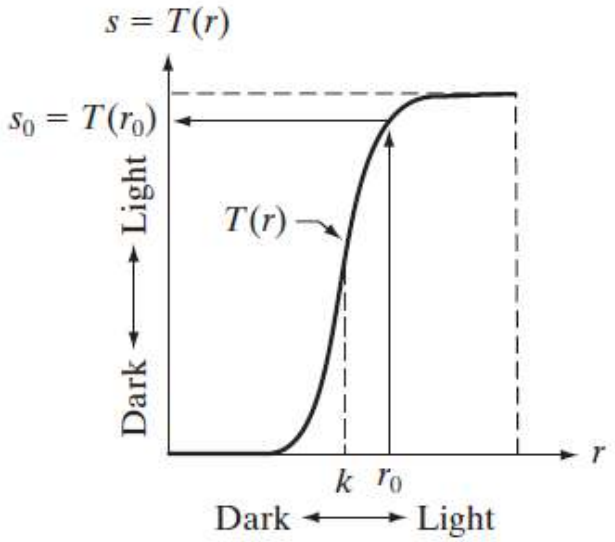
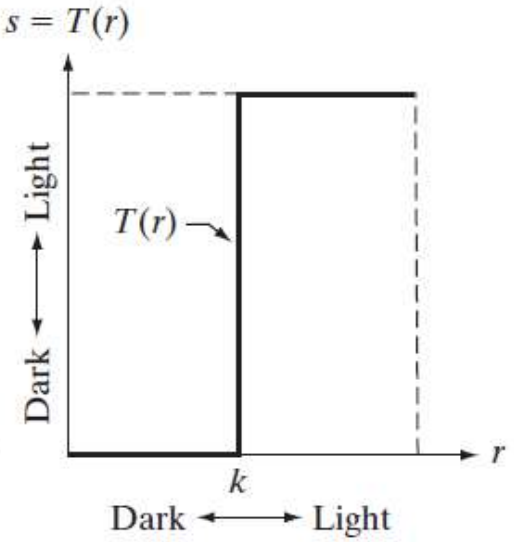


Image with a 3 X 3 spatial mask



Contrast stretching
T function



Thresholding
T function

❖ Contrast stretching:

Original Image

Contrast streched Image

❖ Thresholding:

Original Image

Binary(Threshold) Image

Thank You

9

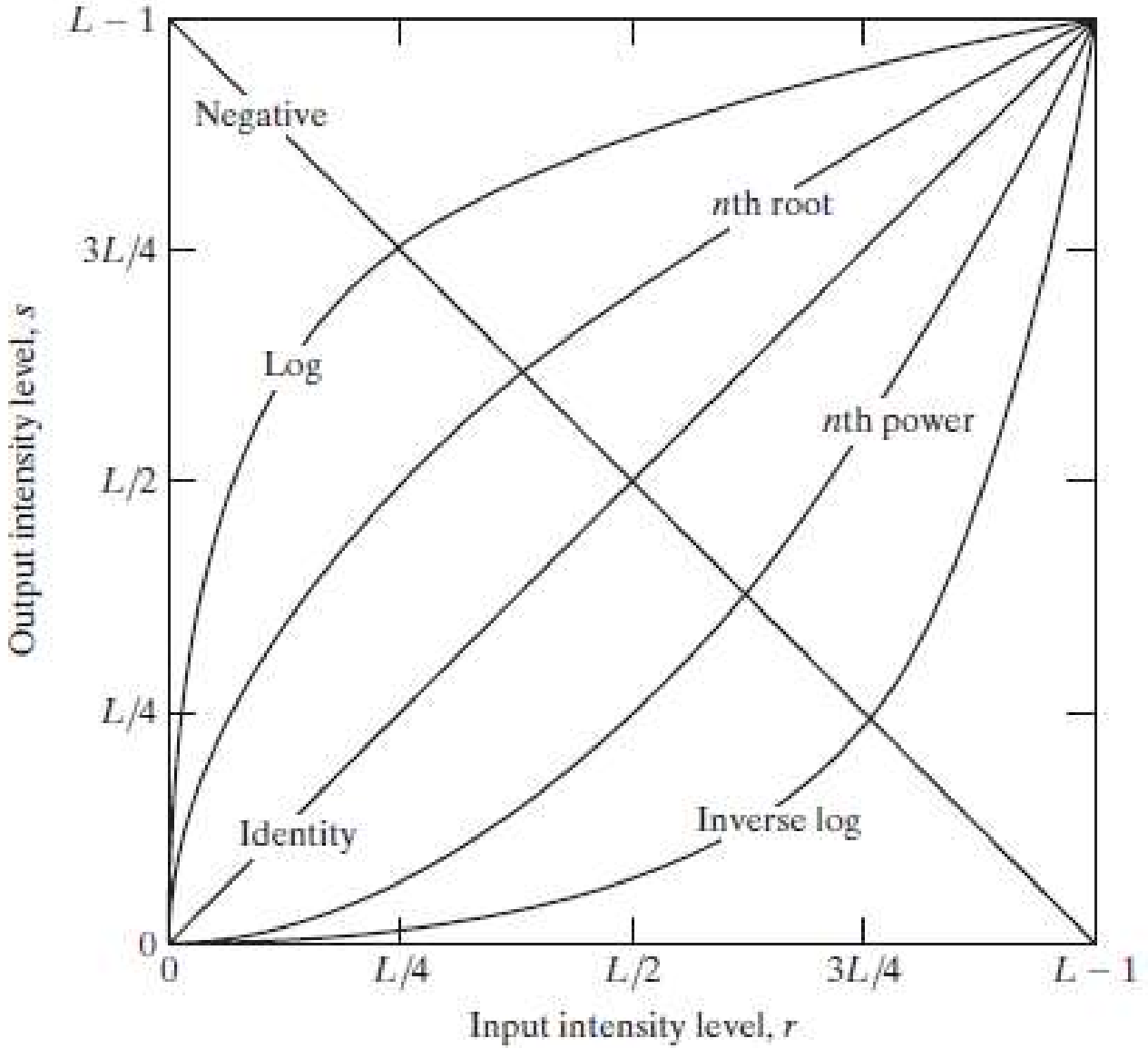
Image negatives, Log and Power-Law transformations for DIP and implementation in MATLAB

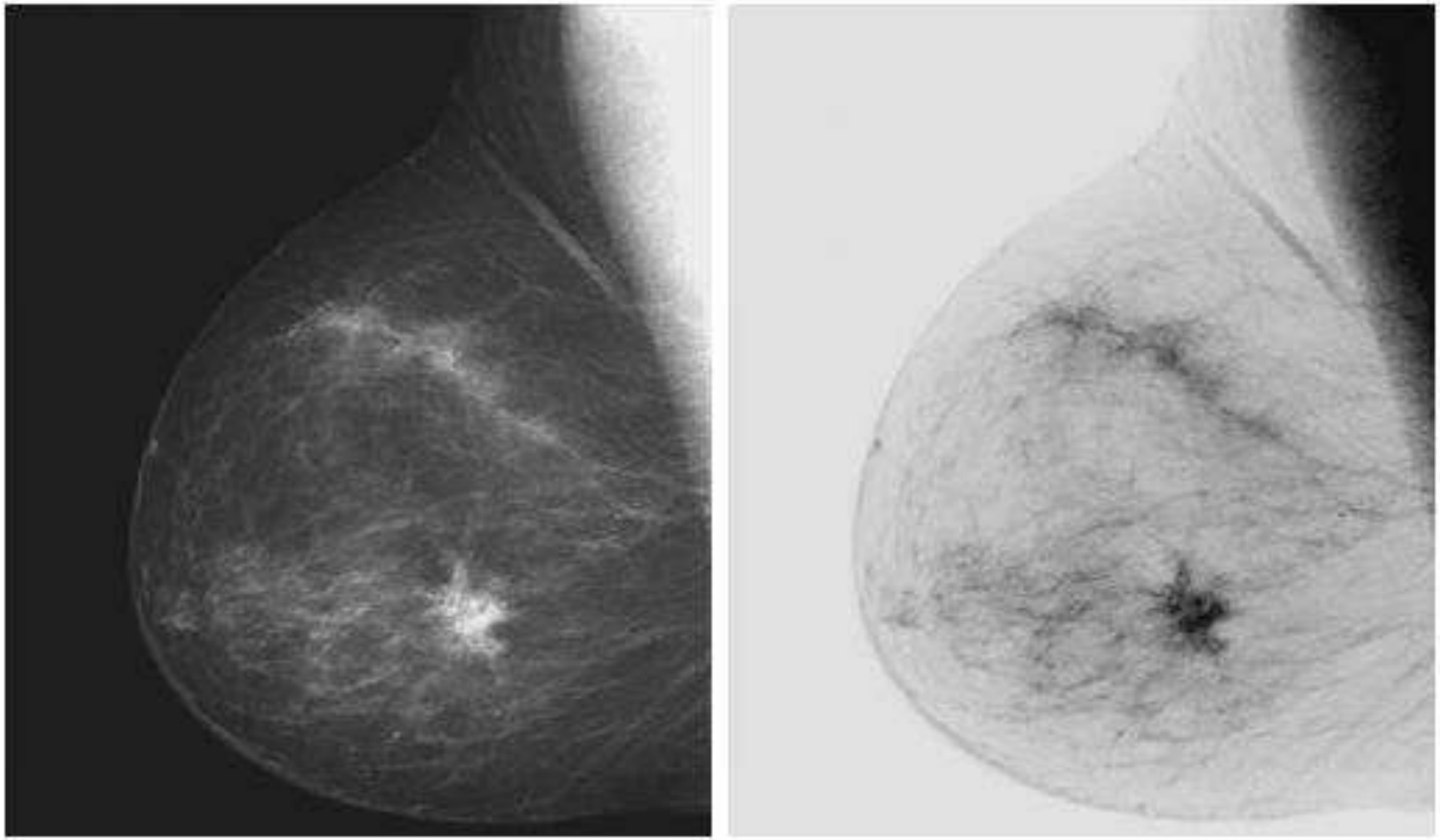
❖ Image Negatives:

- Negative image means inverting the grey levels i.e. black in the original image will become white and vice-versa.
- The transformation for image negative can be given by,

  $s = L - 1 - r$

  where, s is the output intensity value, L is the highest intensity levels and r is the input intensity value
- It is particularly suited for enhancing white or gray detail embedded in dark regions of an image, especially when the black areas are dominant in size


L − 1
3L/4
L/2
L/4
0
Output intensity level, s
Negative
nth root
Log
nth power
Identity
Inverse log
0
L/4
L/2
3L/4
L − 1
Input intensity level, r



a b

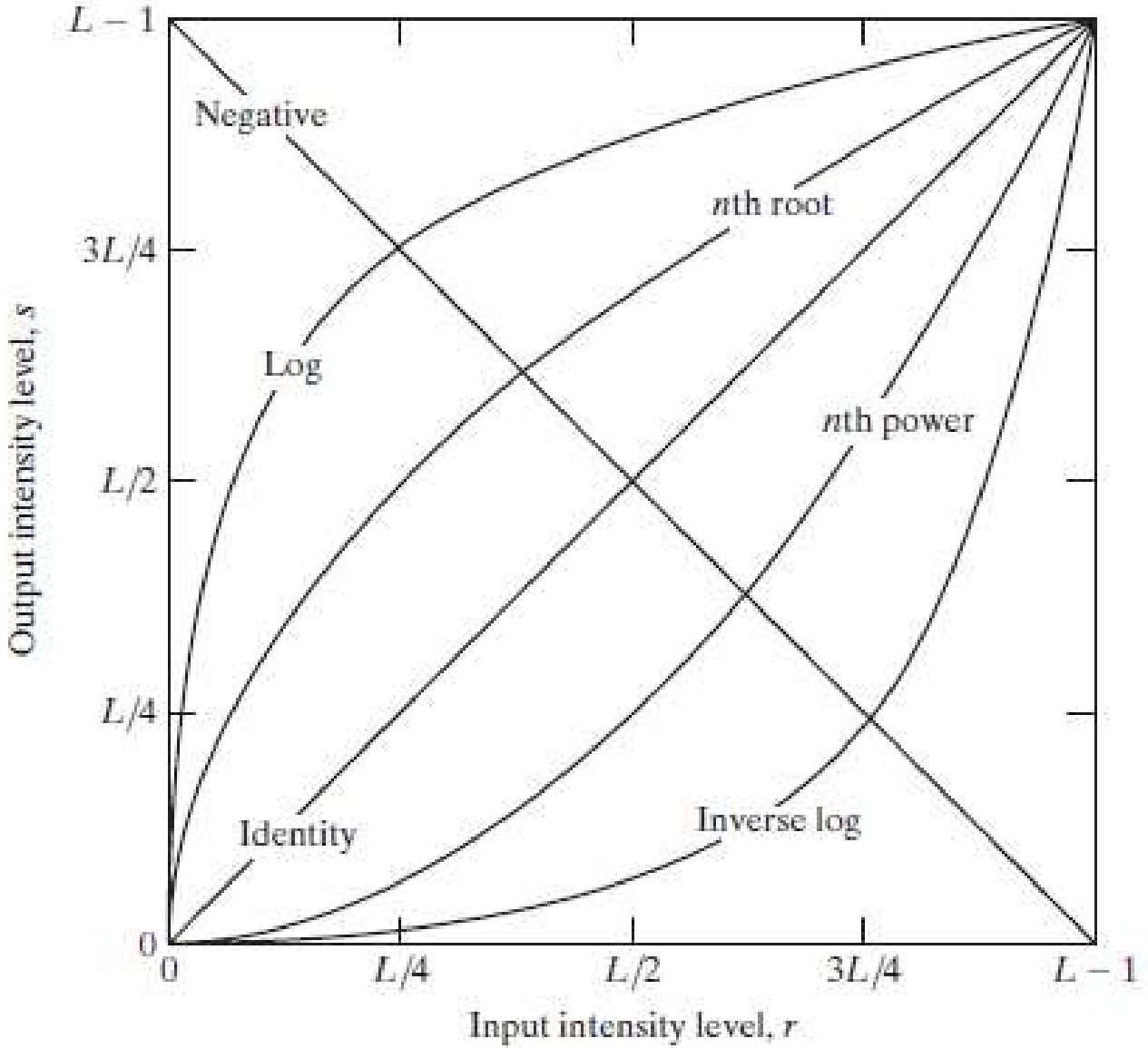
FIGURE 3.4
(a) Original digital mammogram. (b) Negative image obtained using the negative transformation in Eq. (3.2-1). (Courtesy of G.E. Medical Systems.)

Original Image

Negative Image

❖ Log transformations:

- $s = c \log(1 + r)$
  where, c is constant, s is the output intensity value and r is the input intensity value
- It maps a narrow range of low intensity values in the input into a wide range of output levels.
- The opposite is true of higher values of input levels.
- It expands the values of dark pixels in an image while compressing the higher level values.
- It compresses the dynamic range of images with large variations in pixel values.
- Log reduces contrast of brighter regions.


L − 1
3L/4
L/2
L/4
0
Output intensity level, s
Negative
nth root
Log
nth power
Identity
Inverse log
0
L/4
L/2
3L/4
L − 1
Input intensity level, r

Original Image

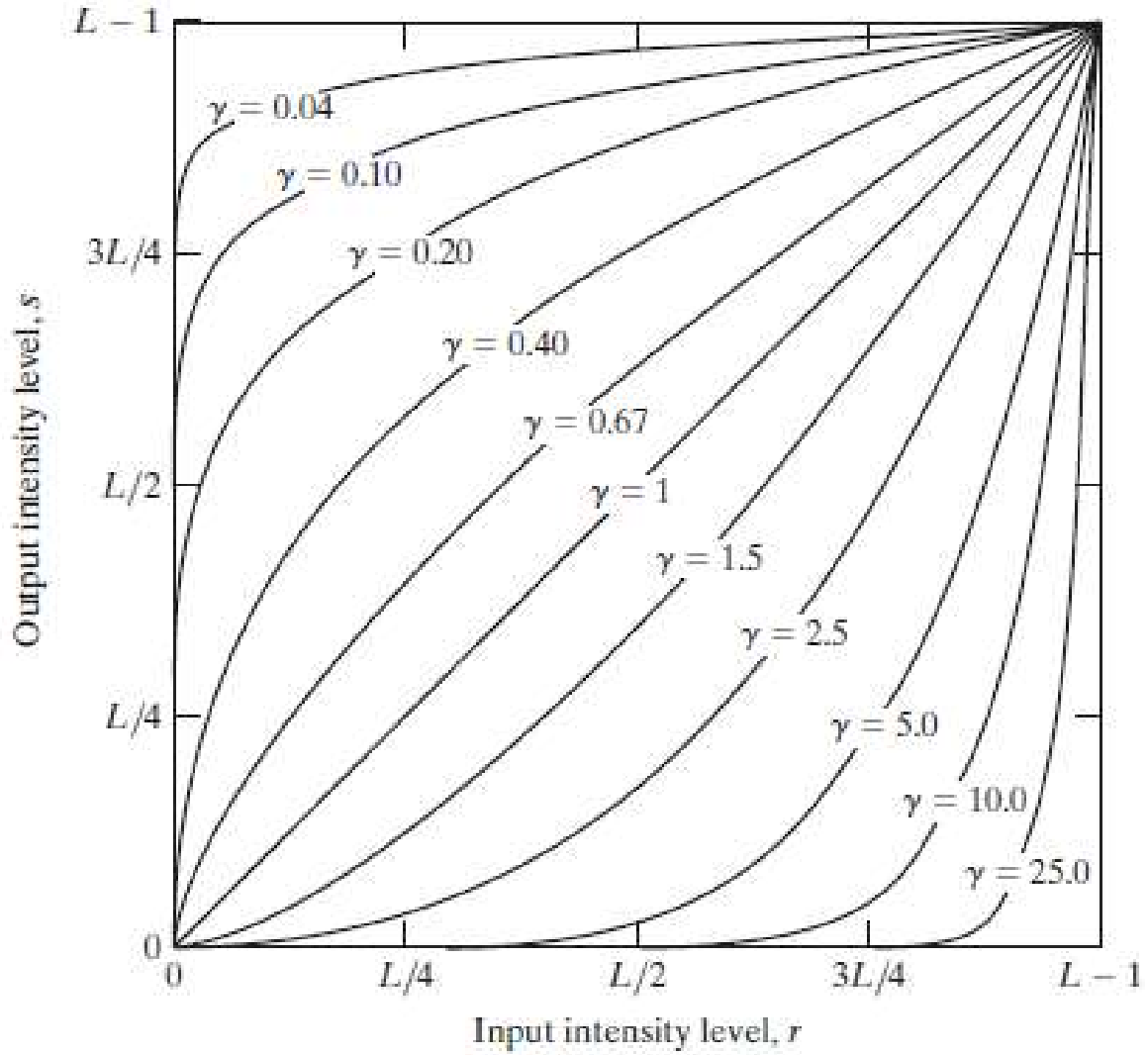
Log factor 2

Log factor 3

Log factor 4

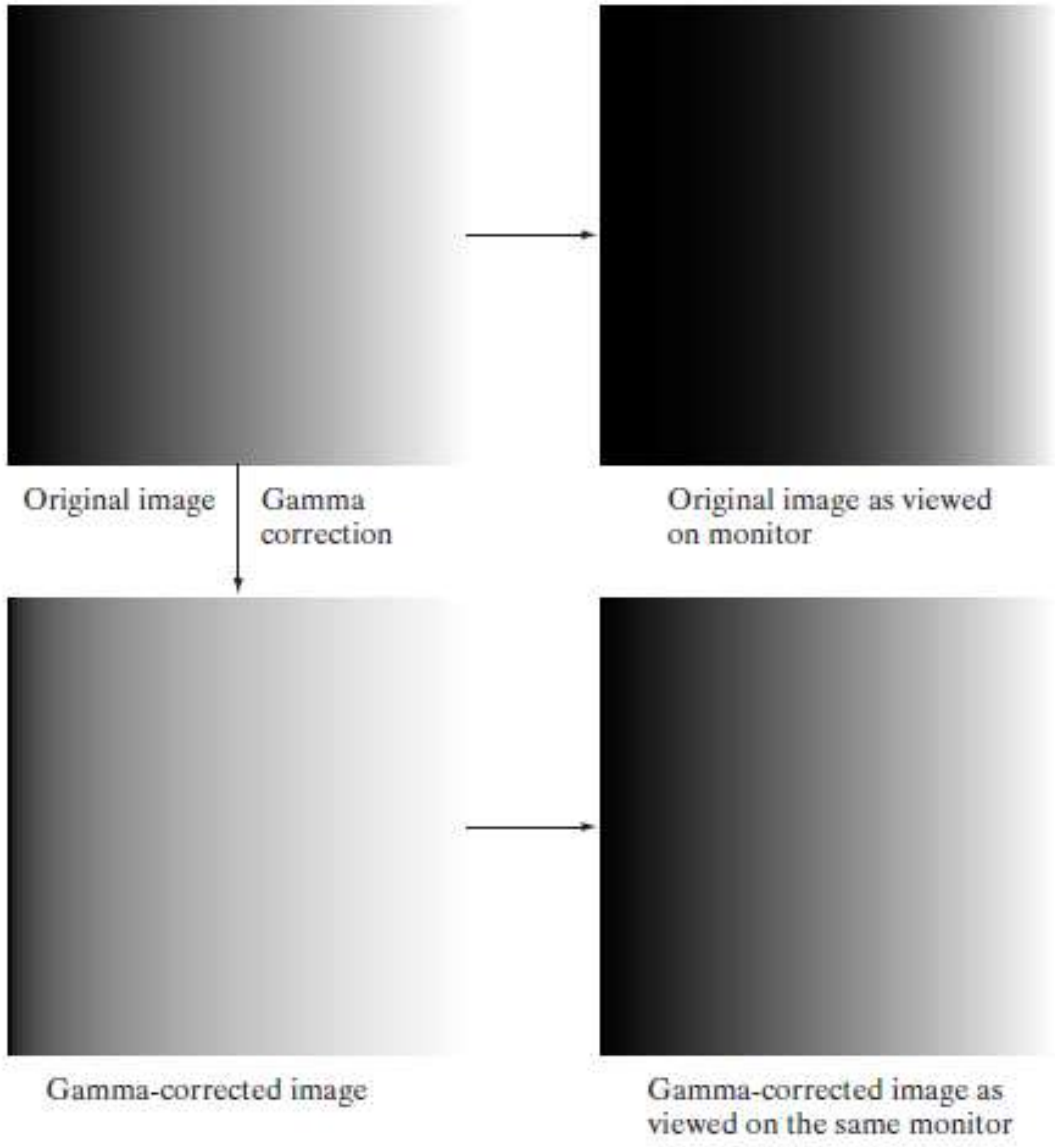
❖ Power-Law (Gamma) transformations:

- $s = c\, r^{\gamma}$
  where c and γ are both positive constants, r =input and s = output
- With fractional values($0<\gamma<1$) of gamma map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher values ($\gamma > 1$)of input levels.
- C=gamma=1 means it is an identity transformations.
- Variety of devices used for image capture , printing, and display respond according to a power law.
- Process used to correct these power law response phenomena is called gamma correction.
- Power-law enhances contrast of brighter regions.


L − 1
3L/4
L/2
L/4
0
0
L/4
L/2
3L/4
L − 1
Output intensity level, s
Input intensity level, r
γ = 0.04
γ = 0.10
γ = 0.20
γ = 0.40
γ = 0.67
γ = 1
γ = 1.5
γ = 2.5
γ = 5.0
γ = 10.0
γ = 25.0

a b
c d

FIGURE 3.7
(a) Intensity ramp image. (b) Image as viewed on a simulated monitor with a gamma of 2.5. (c) Gamma-corrected image. (d) Corrected image as viewed on the same monitor. Compare (d) and (a).



a b
c d

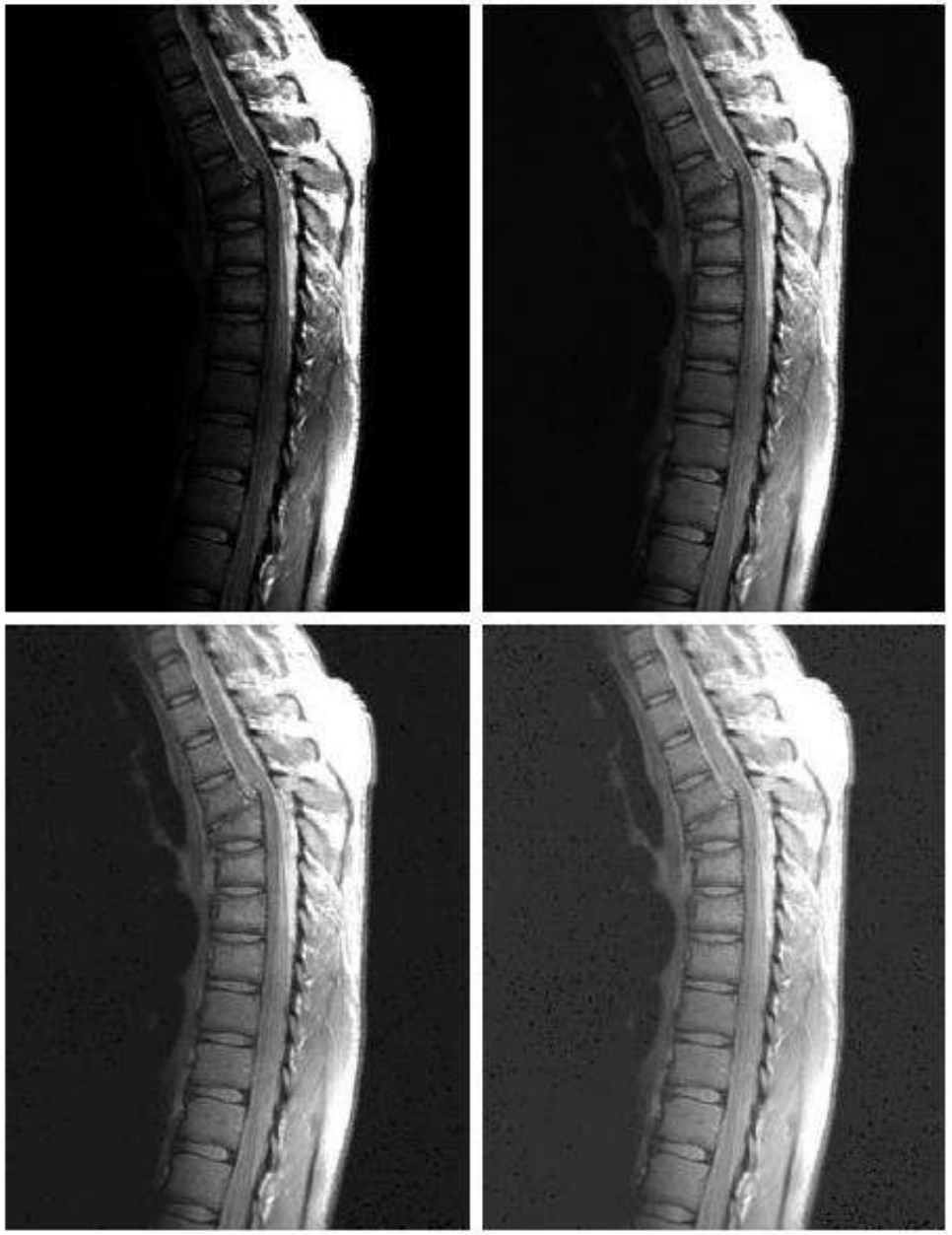
FIGURE 3.9
(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 3.0$, 4.0, and 5.0, respectively. (Original image for this example courtesy of NASA.)

Original Image

Gamma factor 3

Gamma factor 4

Gamma factor 5



a b
c d

FIGURE 3.8
(a) Magnetic resonance image (MRI) of a fractured human spine.
(b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 0.6$, 0.4, and 0.3, respectively. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

Original Image

Gamma factor 0.6

Gamma factor 0.4

Gamma factor 0.3

Thank You

10

Piecewise linear transformation function: Contrast stretching in DIP and implementation in MATLAB

❖ Piece-wise Linear Transformation:

- Piece-wise Linear Transformation is type of gray level transformation that is used for image enhancement. It is a spatial domain method. It is used for manipulation of an image so that the result is more suitable than the original for a specific application.
- Some commonly used piece-wise linear transformations are:

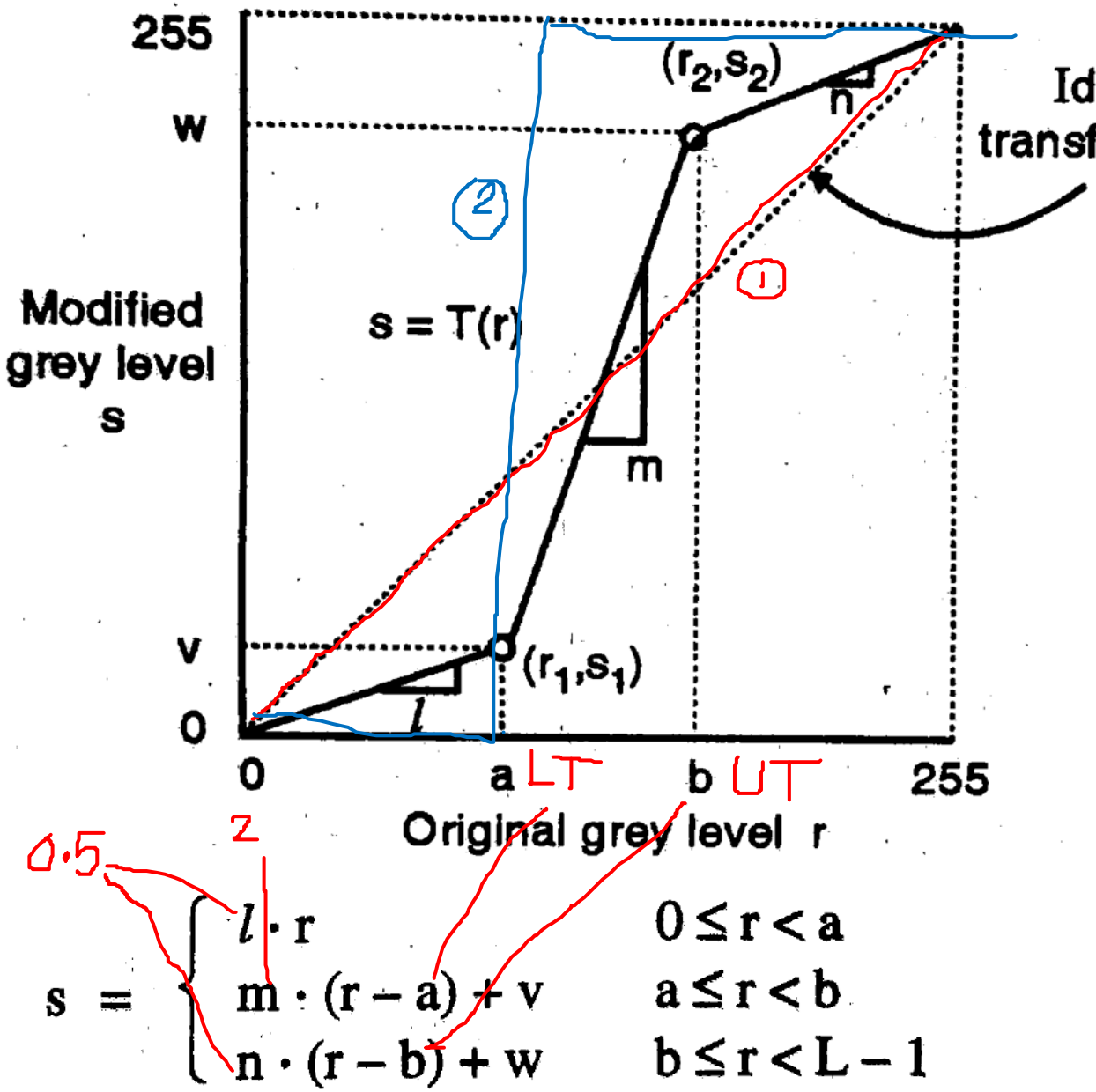
(1) Contrast Stretching

(2) Intensity Level Slicing

(3) Bit Plane Slicing

❖ Contrast Stretching:

- Contrast is the difference between the highest grey level and low grey level of an image.
- Low contrast images can result from poor illuminations, Lack of dynamic range in the imaging sensor, or even the wrong setting of a lens aperture during image acquisition.
- It expands the range of intensity levels in an image so that it spans the full intensity range of display devices.
- The contrast can be stretched by making darker portion more darker and brighter portion more brighter.



$$s = \left(\frac{s_{max} - s_{min}}{r_{max} - r_{min}}\right) * (r - r_{min}) + s_{min}$$

For the first part,

$$s = (0.5) * (r - 0) + 0$$

For the second part,

$$s = (2) * (r - LT) + (0.5 * LT)$$

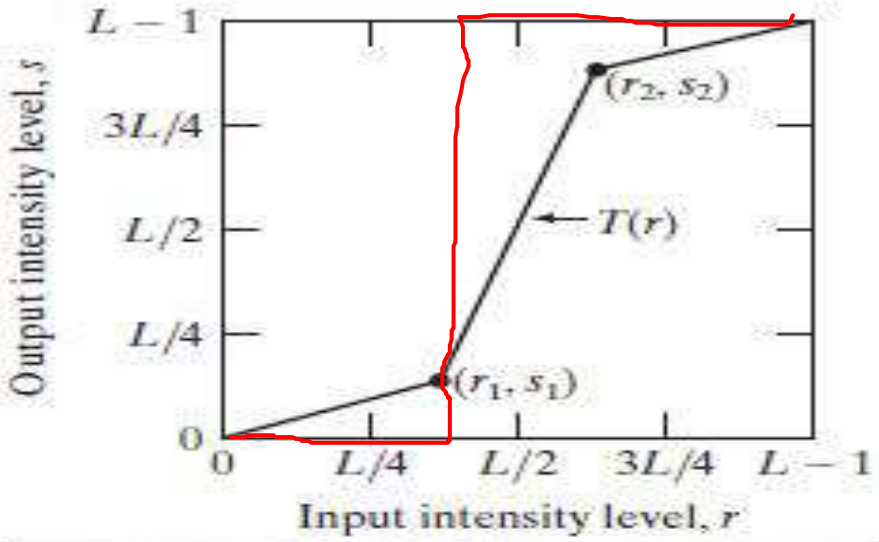
For the third part,

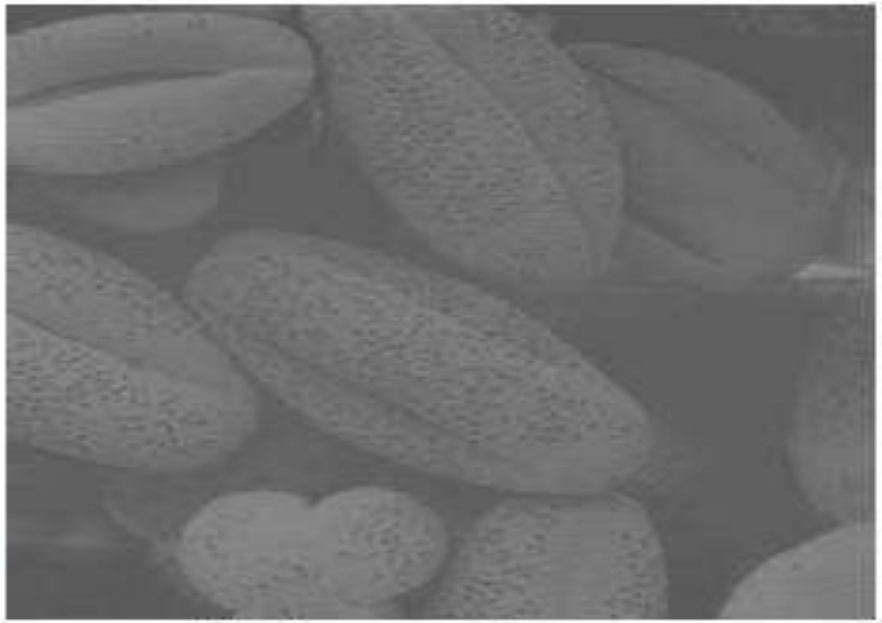
$$s = (0.5) * (r - UT) + (0.5 * LT) + 2 * (UT - LT)$$

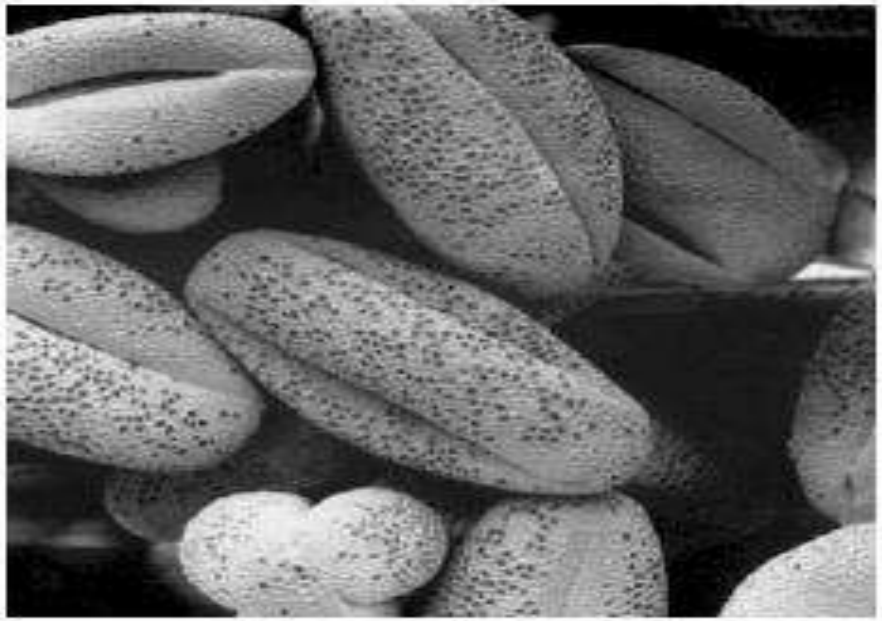
$$s = \begin{cases} l \cdot r & 0 \le r < a \\ m \cdot (r - a) + v & a \le r < b \\ n \cdot (r - b) + w & b \le r < L - 1 \end{cases}$$

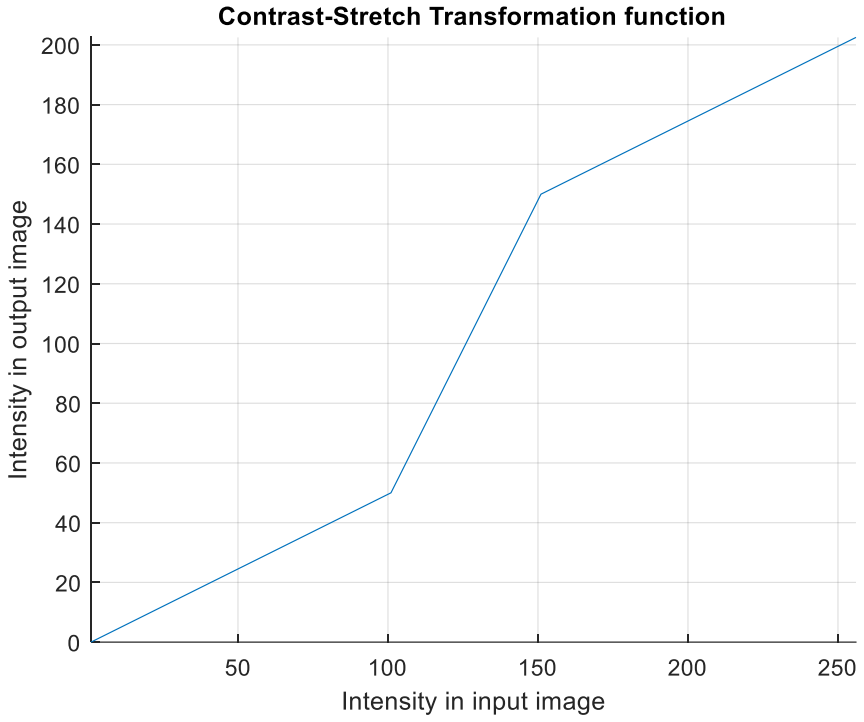
a b
c d

FIGURE 3.10
Contrast stretching. (a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)









Original Image

Contrast stretched Image

Thank You

11

Piecewise linear transformation function: Intensity-level slicing in DIP and implementation in MATLAB

❖ Piece-wise Linear Transformation:

- Piece-wise Linear Transformation is type of gray level transformation that is used for image enhancement. It is a spatial domain method. It is used for manipulation of an image so that the result is more suitable than the original for a specific application.
- Some commonly used piece-wise linear transformations are:

(1) Contrast Stretching

(2) Intensity Level Slicing

(3) Bit Plane Slicing

❖ Intensity/Grey Level Slicing:

- Highlighting specific range of intensities in an image.

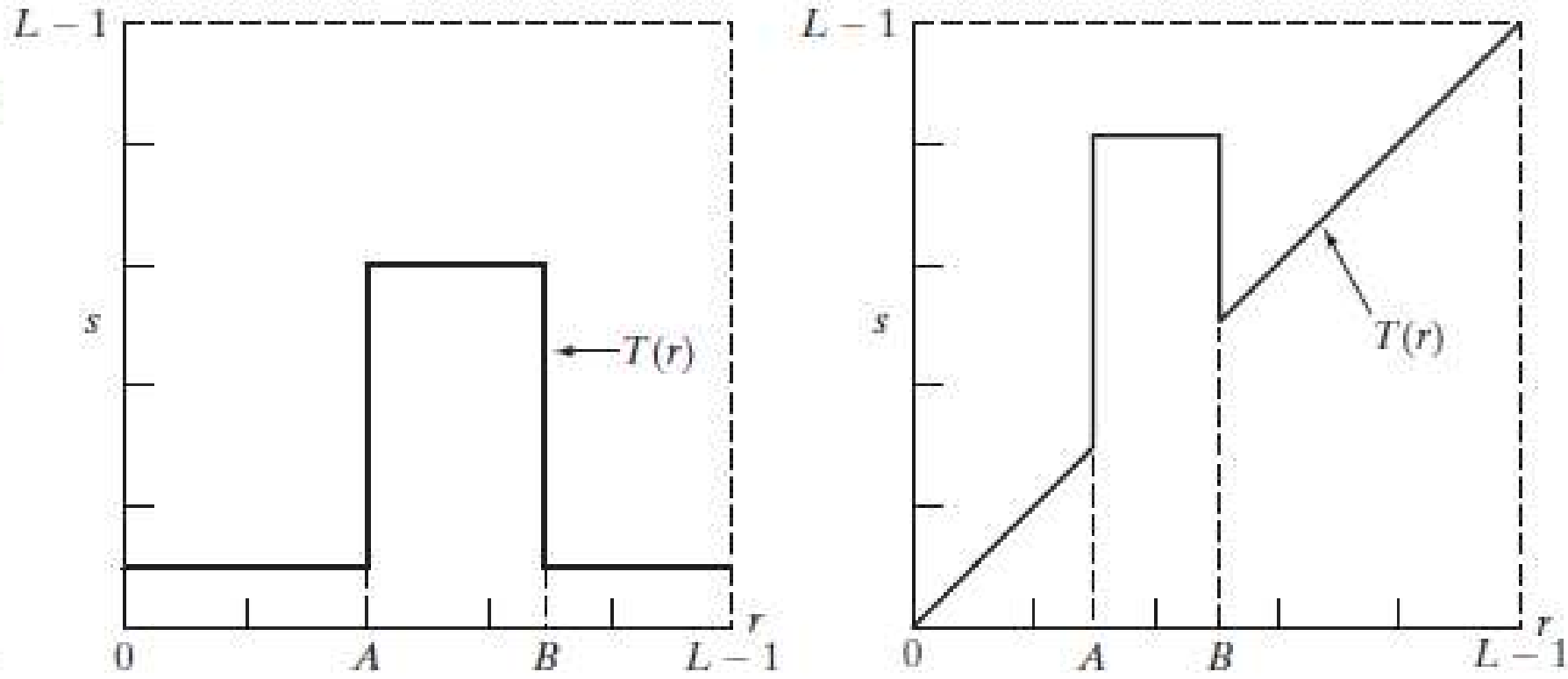
- Enhances features such as masses of water in satellite imagery and enhancing flaws in X-ray images.

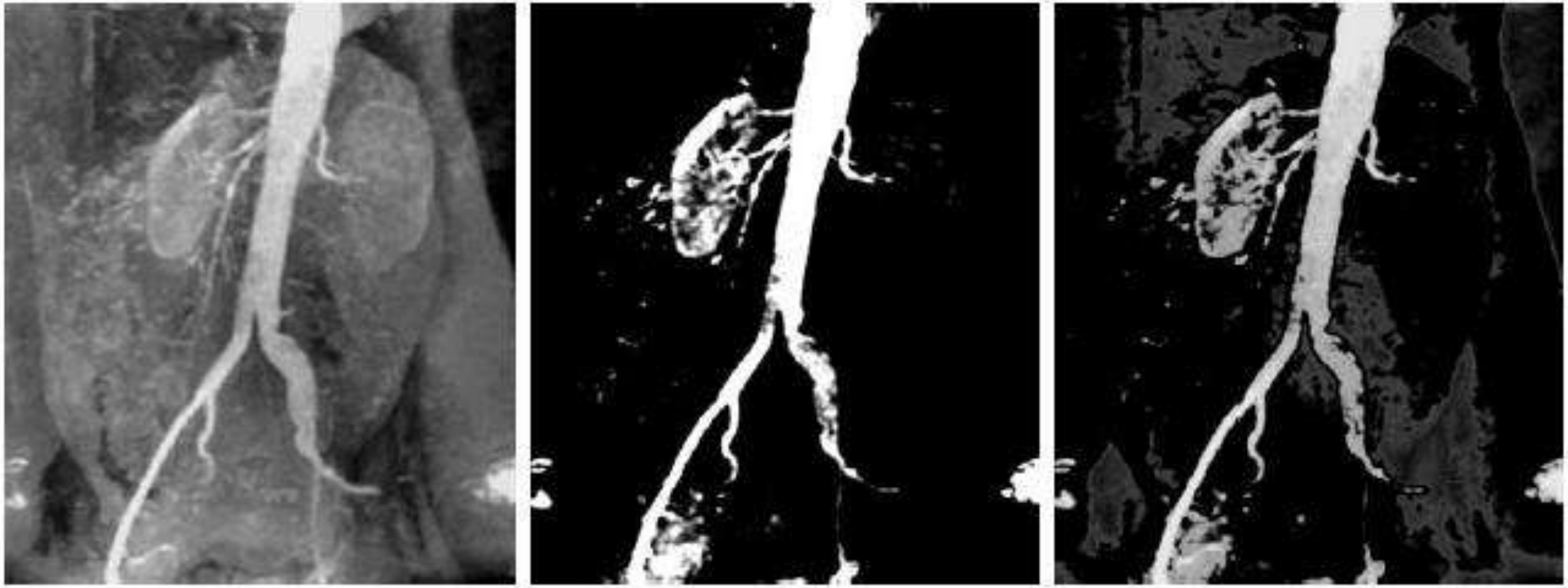
- It can be Implemented two ways:

1) To display only one value (say, white) in the range of interest and rests are black which produces binary image.
2) Brightens (or darkens) the desired range of intensities but leaves all other intensity levels in the image unchanged.

a b

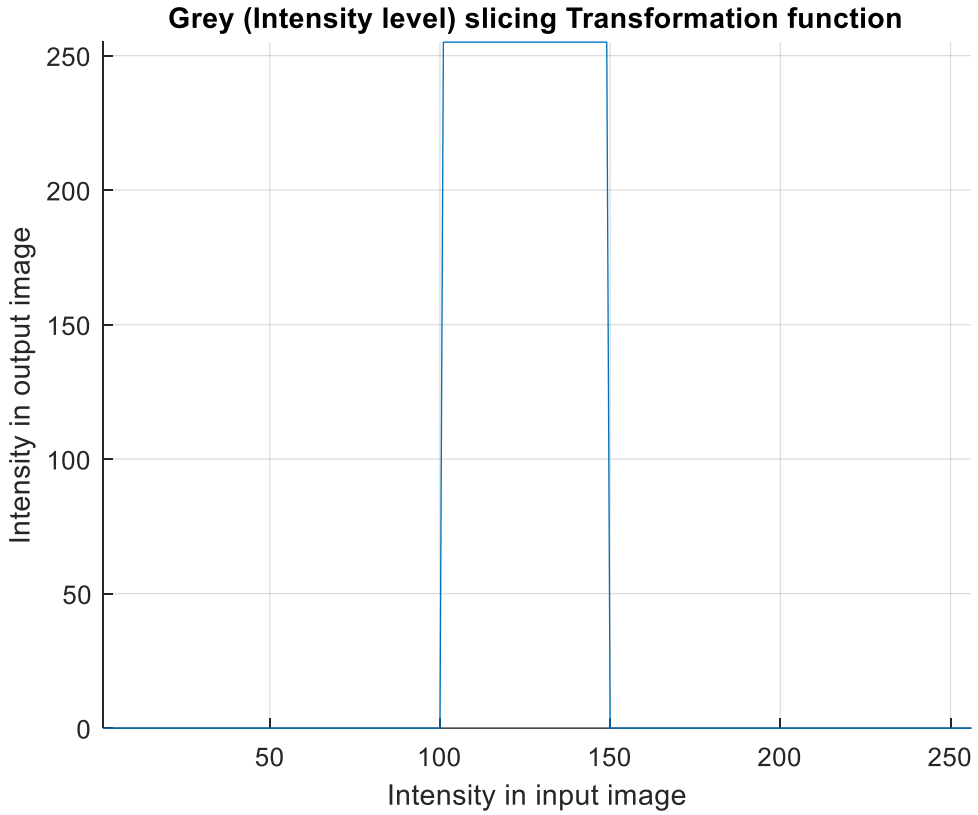
FIGURE 3.11 (a) This transformation highlights intensity range $[A, B]$ and reduces all other intensities to a lower level. (b) This transformation highlights range $[A, B]$ and preserves all other intensity levels.



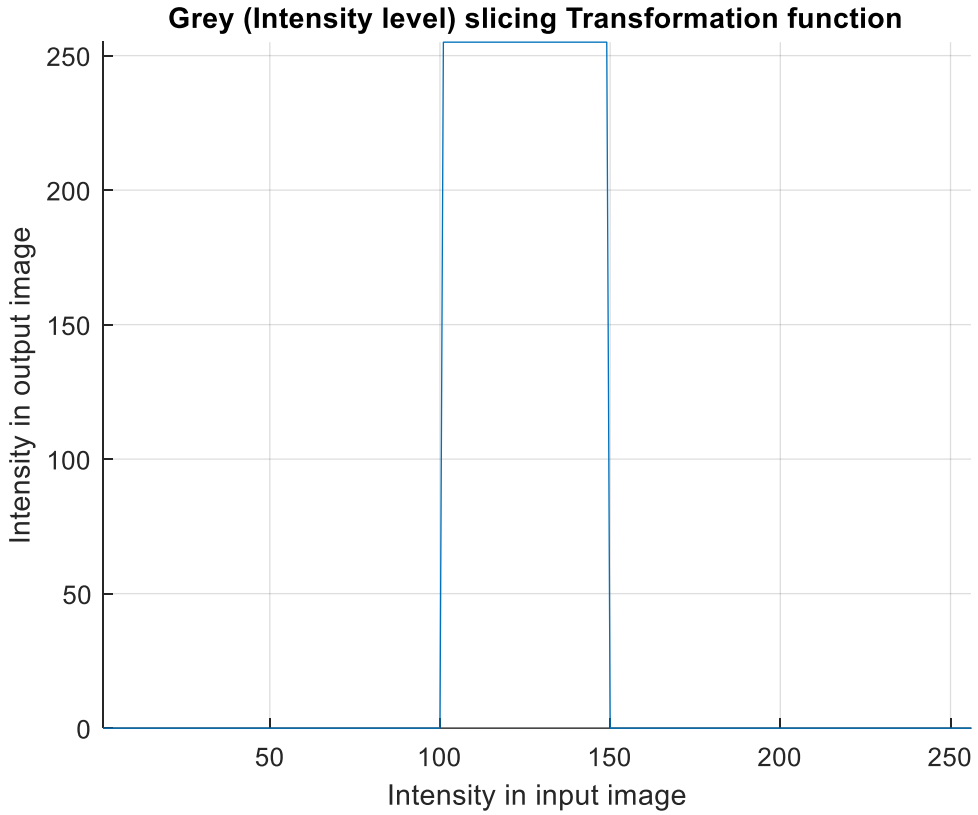


a b c

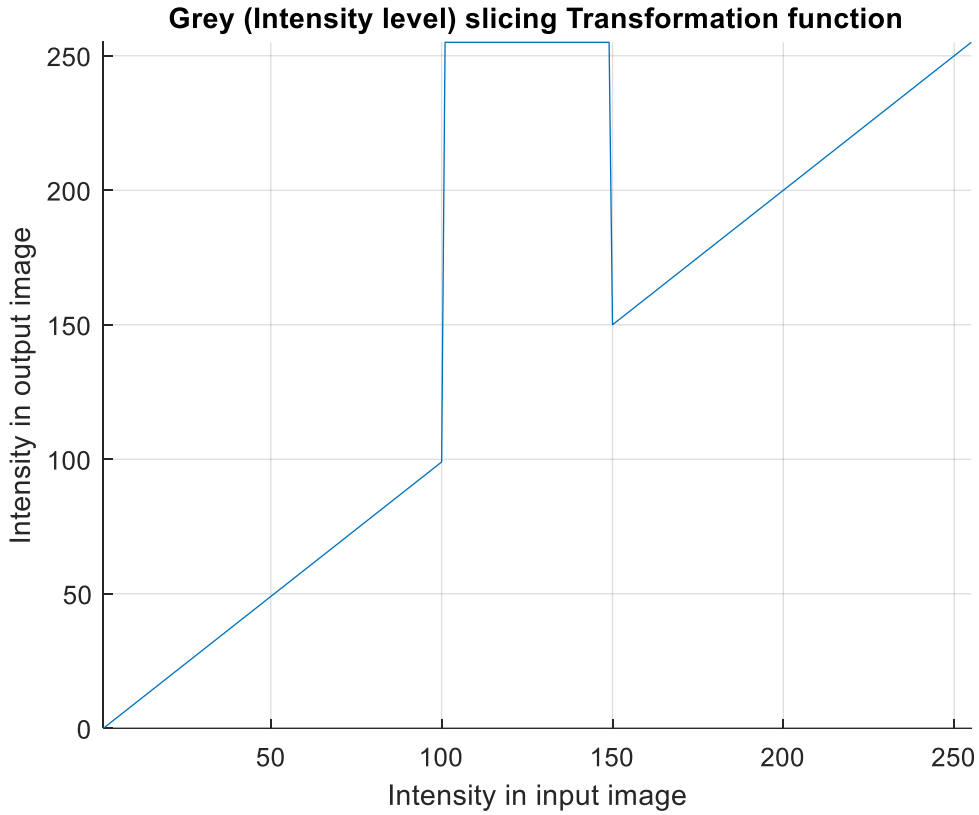
FIGURE 3.12 (a) Aortic angiogram. (b) Result of using a slicing transformation of the type illustrated in Fig. 3.11(a), with the range of intensities of interest selected in the upper end of the gray scale. (c) Result of using the transformation in Fig. 3.11(b), with the selected area set to black, so that grays in the area of the blood vessels and kidneys were preserved. (Original image courtesy of Dr. Thomas R. Gest, University of Michigan Medical School.)


Grey (Intensity level) slicing Transformation function
Intensity in output image
Intensity in input image
0
50
100
150
200
250
50
100
150
200
250

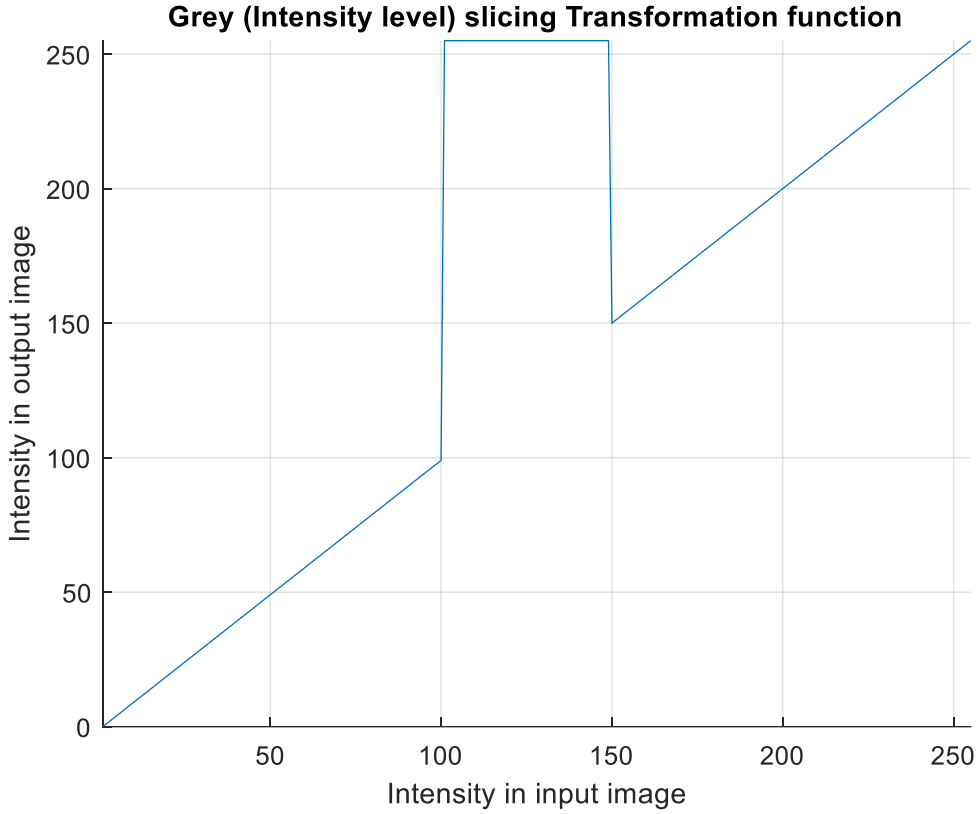

Original Image
Intensity sliced Image
Amul


Grey (Intensity level) slicing Transformation function
Intensity in output image
Intensity in input image
0
50
100
150
200
250
50
100
150
200
250


Original Image
Intensity sliced Image


Grey (Intensity level) slicing Transformation function
Intensity in output image
Intensity in input image
0
50
100
150
200
250


Original Image
Intensity sliced Image
Amul
Amul


Grey (Intensity level) slicing Transformation function
Intensity in output image
Intensity in input image
0
50
100
150
200
250
50
100
150
200
250


Original Image
Intensity sliced Image

Thank You

12

Piecewise linear transformation function: Bit-plane slicing in DIP and implementation in MATLAB

❖ Piece-wise Linear Transformation:

- Piece-wise Linear Transformation is type of gray level transformation that is used for image enhancement. It is a spatial domain method. It is used for manipulation of an image so that the result is more suitable than the original for a specific application.
- Some commonly used piece-wise linear transformations are:

(1) Contrast Stretching

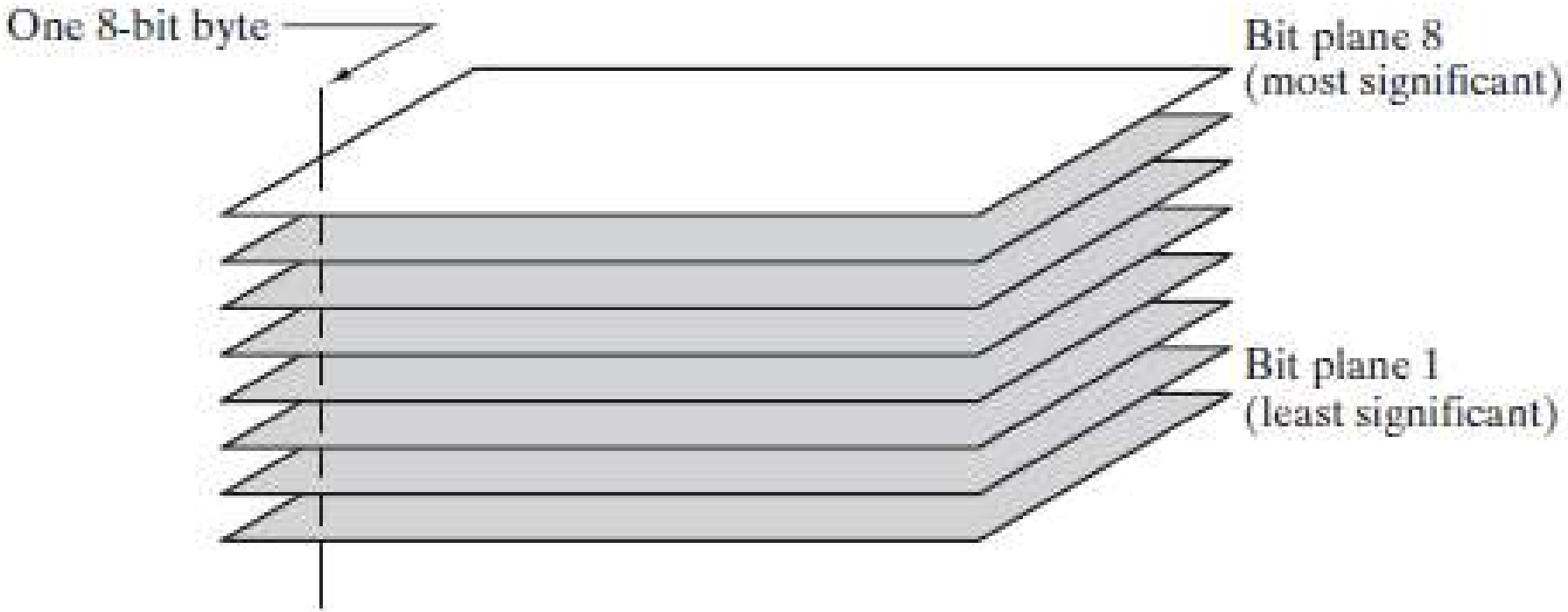
(2) Intensity Level Slicing

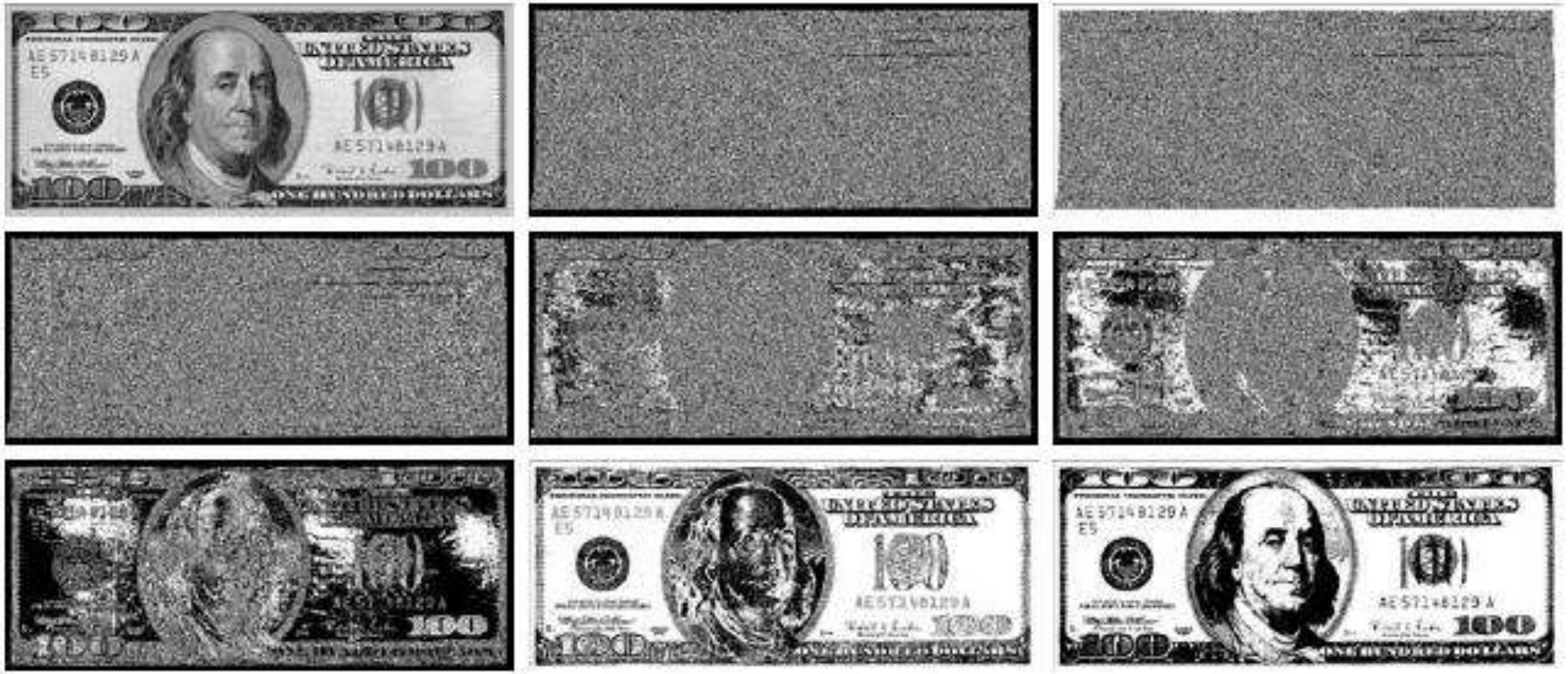
(3) Bit Plane Slicing

(3) Bit Plane Slicing:

- Pixels are digital numbers composed of bits.
- 256 gray scale image is composed of 8 bits.
- Instead of highlighting intensity level ranges, we could highlight the contribution made to total image appearance by specific bits.
- 8-bit image may be considered as being composed of eight 1-bit planes, with plane 1 containing the lowest order bit of all pixels in the image and plane 8 all the highest-order bits.

FIGURE 3.13
Bit-plane representation of an 8-bit image.



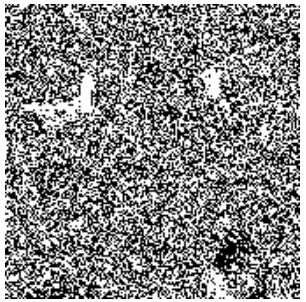


a b c
d e f
g h i

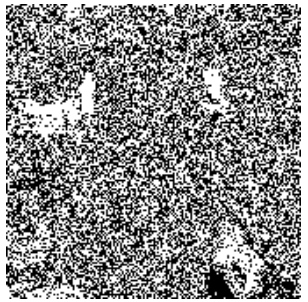
FIGURE 3.14 (a) An 8-bit gray-scale image of size 500 × 1192 pixels. (b) through (i) Bit planes 1 through 8, with bit plane 1 corresponding to the least significant bit. Each bit plane is a binary image.

Original Image

Bit plane 1



Bit plane 2



Bit plane 3

Bit plane 4

Bit plane 5

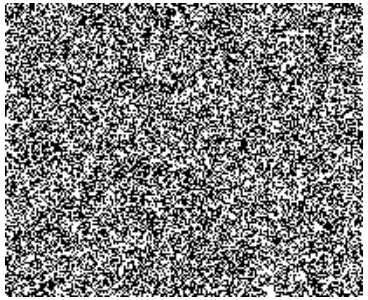
Bit plane 6

Bit plane 7

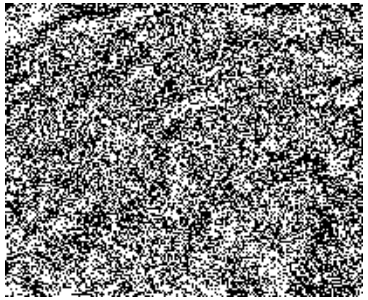
Bit plane 8

Original Image

Bit plane 1

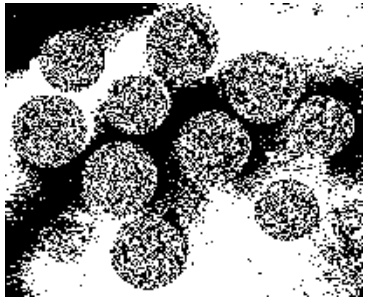


Bit plane 2

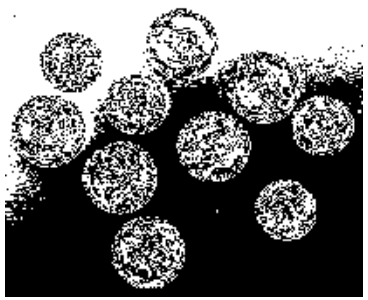


Bit plane 3

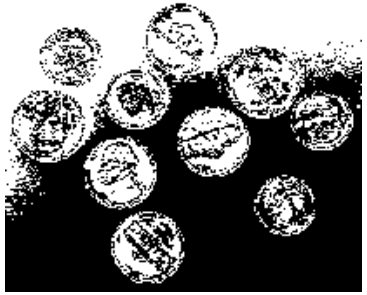
Bit plane 4



Bit plane 5

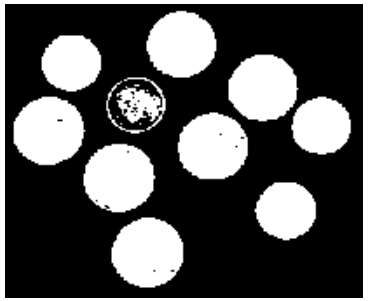


Bit plane 6



Bit plane 7

Bit plane 8



Thank You

13

Histogram Equalization in DIP and its implementation in MATLAB

❖ Histogram equalization:

- Histogram is the graphical representation of any data.
- Histogram provide a global description of the appearance of an image. It is a spatial domain method.
- Histogram is the representation of relative frequency of occurrence of various grey levels of an image.
- Histogram equalization is used for image enhancement.
- Histogram can control the quality of an image by normalizing the histogram values to a flat profile.

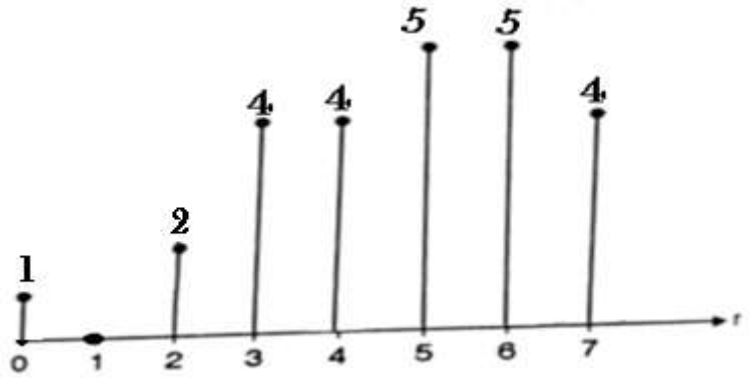
Image

Gray level

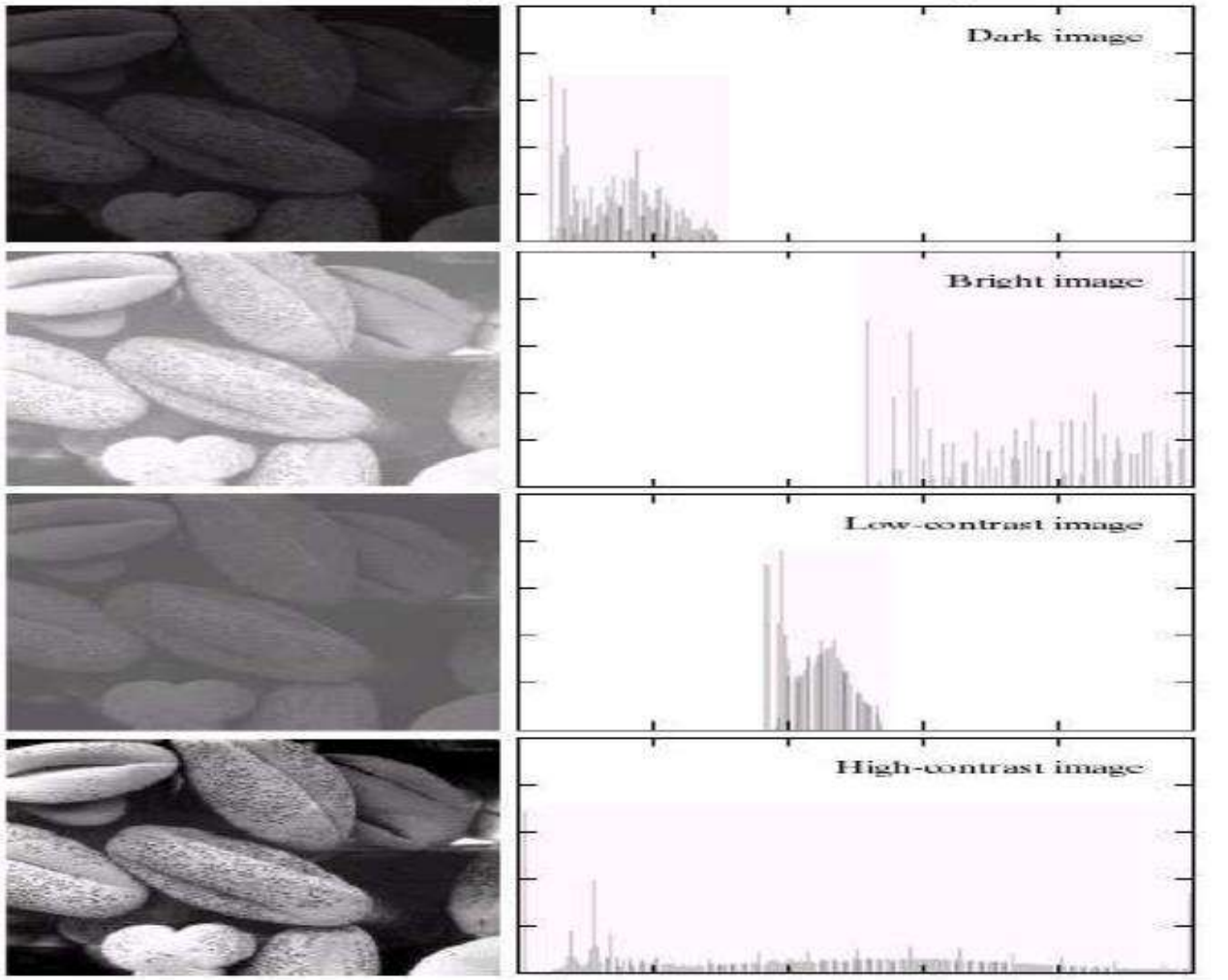
3	3	7	7	6
5	2	2	3	4
6	6	4	4	5
5	7	3	6	0
7	6	5	5	4

$2^3 = 8$

Gray levels	0	1	2	3	4	5	6	7
Frequency of occurance (no. of pixels)	1	0	2	4	4	5	5	4

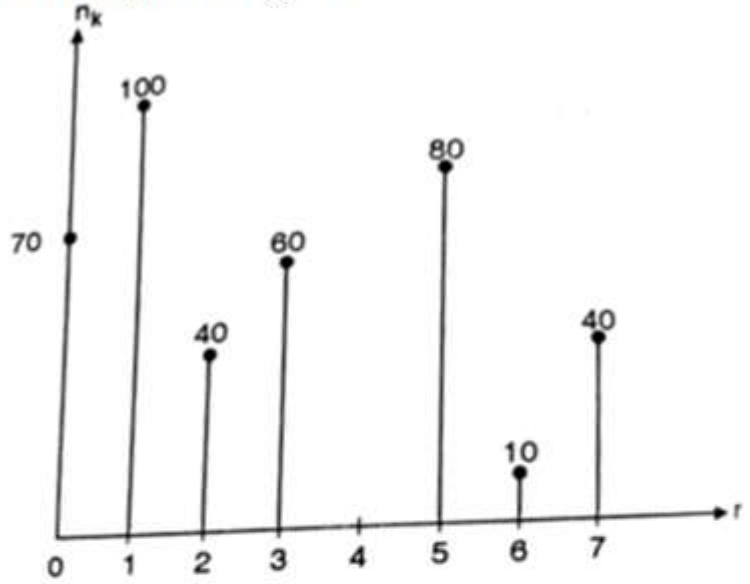


Histogram Processing



❖ Steps in Histogram processing

Step 1: Plot the original Histogram



Step 2 : Perform Histogram Equalization

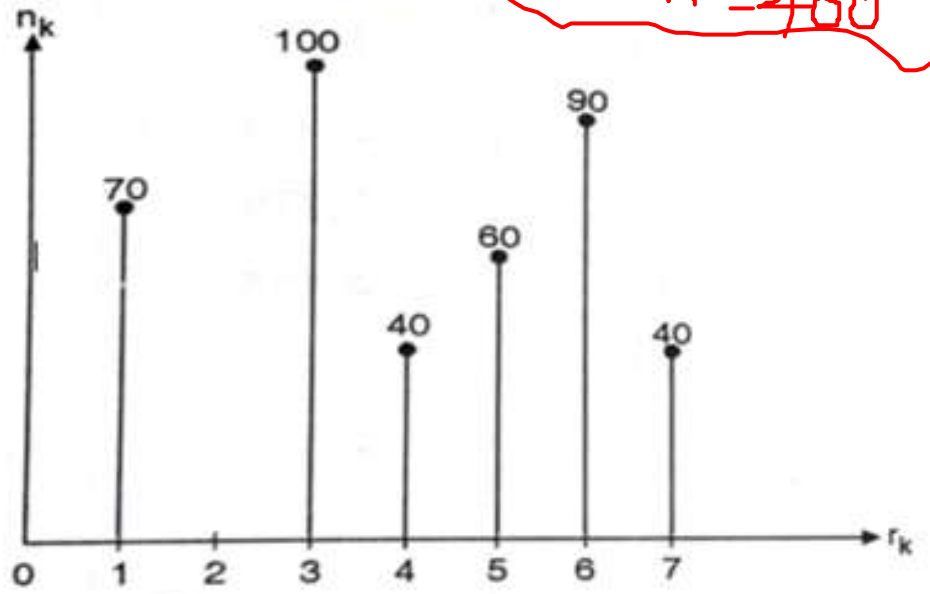
Grey Level	n_k	PDF=n_k/n	CDF=$\sum PDF$	CDF x (L-1) CDF x 7	Rounding off (New Grey Level)
0	70	0.175	0.175	1.22	1
1	100	0.25	0.425	2.97	3
2	40	0.1	0.525	3.67	4
3	60	0.15	0.675	4.72	5
4	0	0	0.675	4.72	5
5	80	0.2	0.875	6.12	6
6	10	0.025	0.9	6.3	6
7	40	0.1	1	7	7
Total	n=400				

Step 3: Make new histogram table

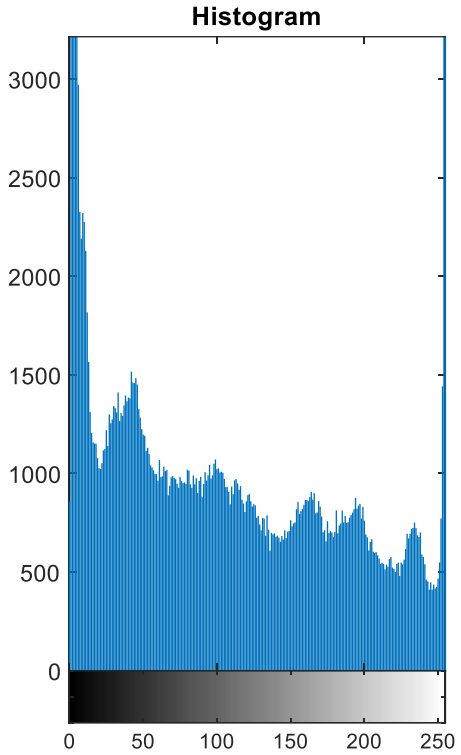
Old Grey Level	n_k	New Grey Level	New n_k
0	70	1	70
1	100	3	100
2	40	4	40
3	60	5	60 + 0 = 60
4	0	5	
5	80	6	80 + 10 = 90
6	10	6	
7	40	7	40

n =400

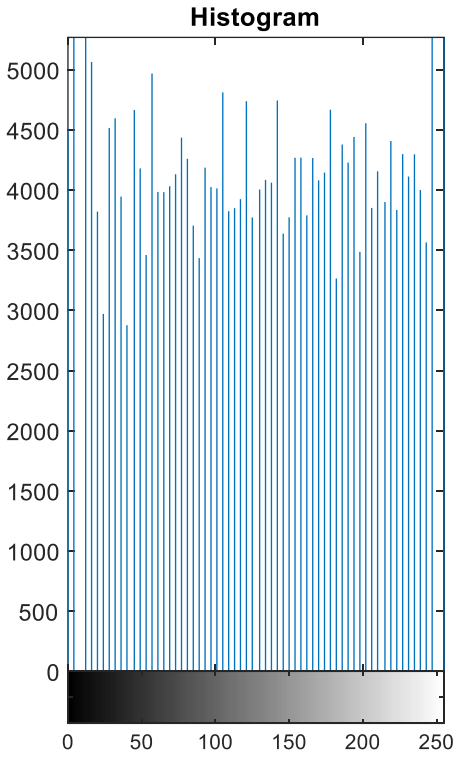
Step 4: Plot the equalized Histogram



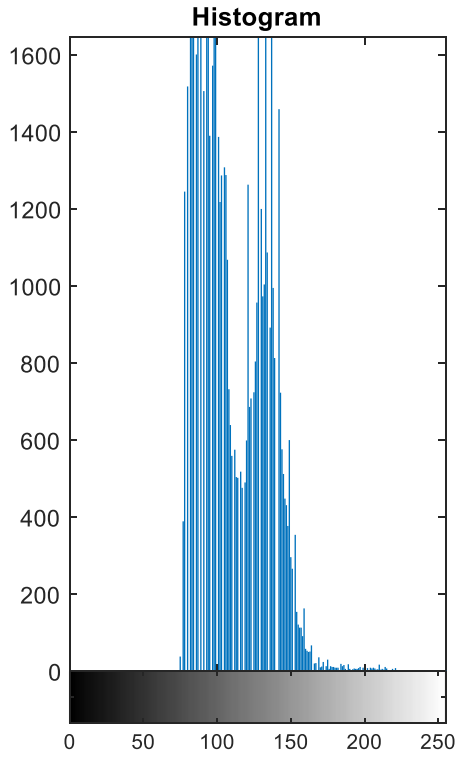
Original image



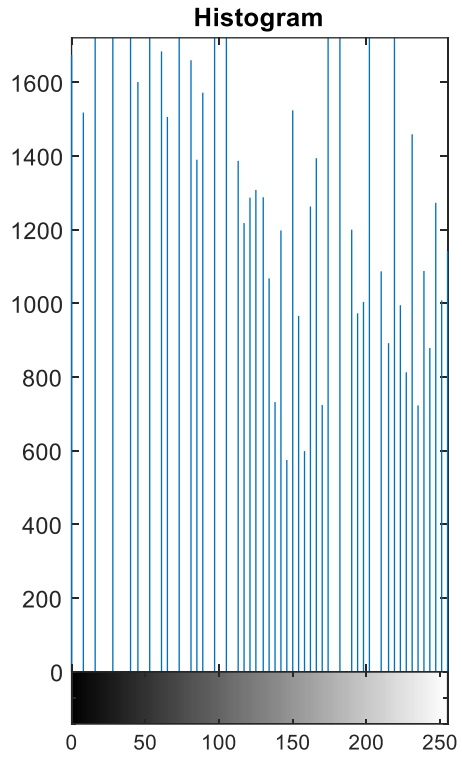
Histogram equalized image



Original image



Histogram equalized image



Thank You

14

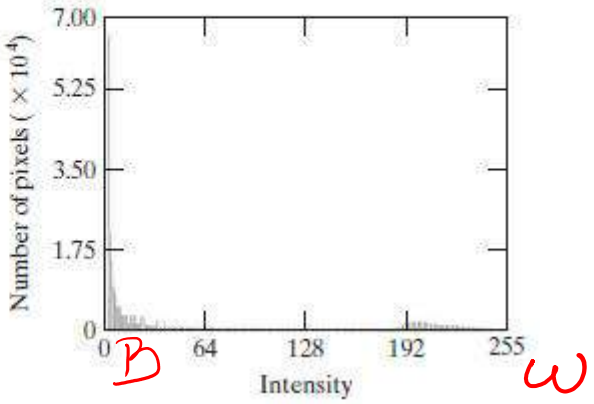
Histogram Specification /Matching in DIP and its implementation in MATLAB

❖ Histogram Matching/Specification:

- Histogram equalization is automatic and is not always good.
- At many times it is desirable to have an interactive method in which certain grey levels are highlighted.
- If we modify the grey level of an image that has a uniform PDF, using the inverse transformation we can get back the original histogram. Using this knowledge, we can obtain any shape of the grey level histogram by processing the given image.
- The method used to generate a processed image that has a specified histogram is called histogram matching or specification.

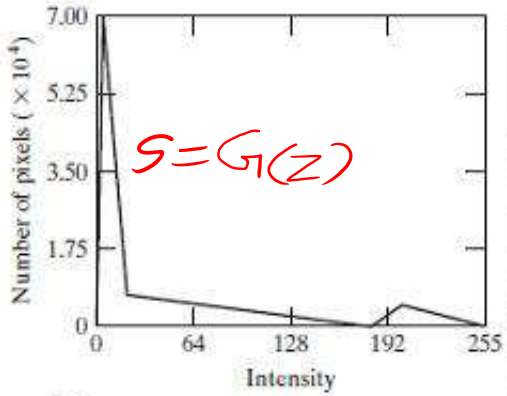
a b

FIGURE 3.23
(a) Image of the Mars moon Phobos taken by NASA's *Mars Global Surveyor*. (b) Histogram. (Original image courtesy of NASA.)

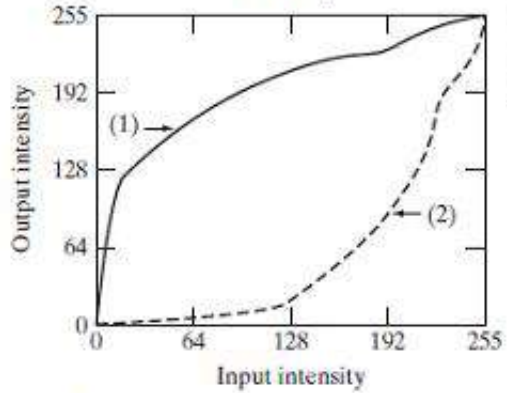


a c
b
d

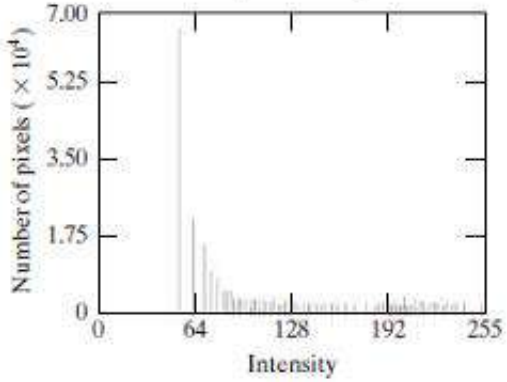
FIGURE 3.25
(a) Specified histogram. (b) Transformations. (c) Enhanced image using mappings from curve (2). (d) Histogram of (c).

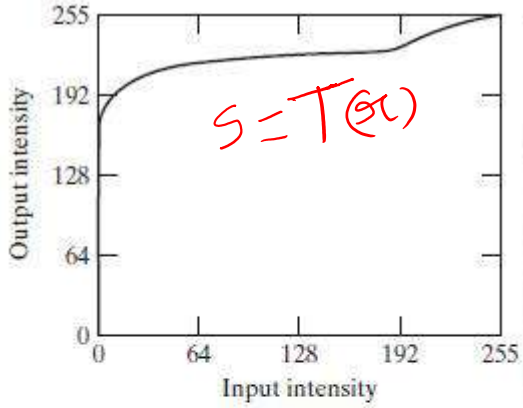


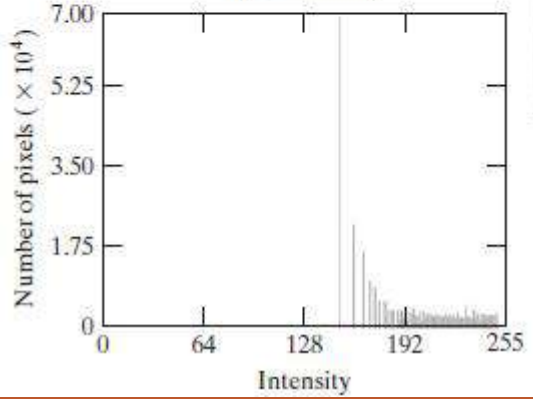
$G(z_q) = s_k$



$z_q = G^{-1}(s_k)$







a b
c

FIGURE 3.24
(a) Transformation function for histogram equalization. (b) Histogram-equalized image (note the washed-out appearance). (c) Histogram of (b).

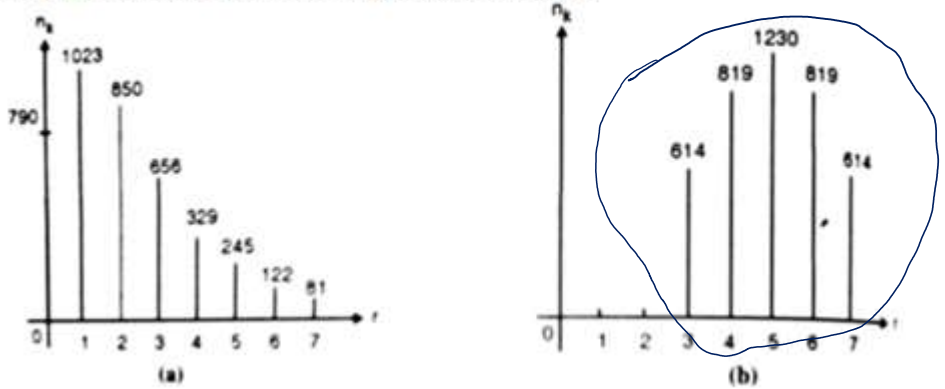
Grey Level	0	1	2	3	4	5	6	7
No. of pixels	790	1023	850	656	329	245	122	81

Image A

Grey Level	0	1	2	3	4	5	6	7
No. of pixels	0	0	0	614	819	1230	819	614

Image B

Step 1 : Plot histogram for image A and Image B



Step 2 : Equalize image A

Grey Level	n_k	$P_r(n_k) = n_k/n$	CDF	CDF x (L-1) L=8	New Grey Level	New Pixel Value
0	790	0.19	0.19	1.33	1	790
1	1023	0.25	0.44	3.08	3	1023
2	850	0.21	0.65	4.55	5	850
3	656	0.16	0.81	5.67	6	656 +
4	329	0.08	0.89	6.23	6	329 =985
5	245	0.06	0.95	6.25	7	245 +
6	122	0.03	0.98	6.86	7	122 + 81
7	81	0.02	1	7	7	= 448
Total (n)	4096					

Step 3 : Equalize image B

Before equalizing image B, note the values of the distinct histogram levels after histogram equalization of image A.

Grey Level	0	1	2	3	4	5	6	7
No. of pixels	0	790	0	1023	0	850	985	448

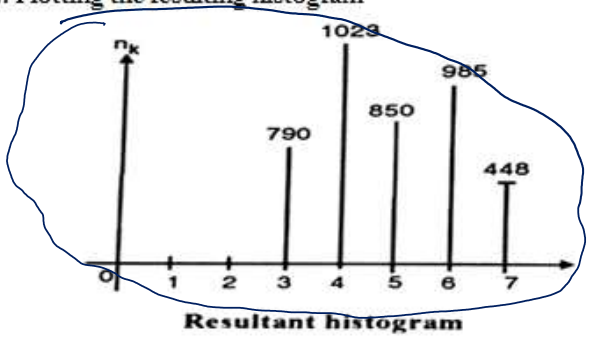
Now equalize image B

Grey Level	n_k	$P_r(n_k) = n_k/n$	CDF	CDF x (L-1) L=8	Rounding off	New Grey Level
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	614	0.149	0.149	1.05	1	1
4	819	0.20	0.35	2.5	3	3
5	1230	0.30	0.65	4.55	5	5
6	819	0.20	0.85	5.97	6	6
7	614	0.15	1	7	7	7
Total(n)	4096					

Grey level	Equalization values of Problem Image	Equalization values of Reference Image	Mapping For Histogram Specification
0	1	0	3
1	3	0	4
2	5	0	5
3	6	1	6
4	6	3	6
5	7	5	7
6	7	6	7
7	7	7	7

790
1023
850
985
448

Step 4: Plotting the resulting histogram



Resultant histogram

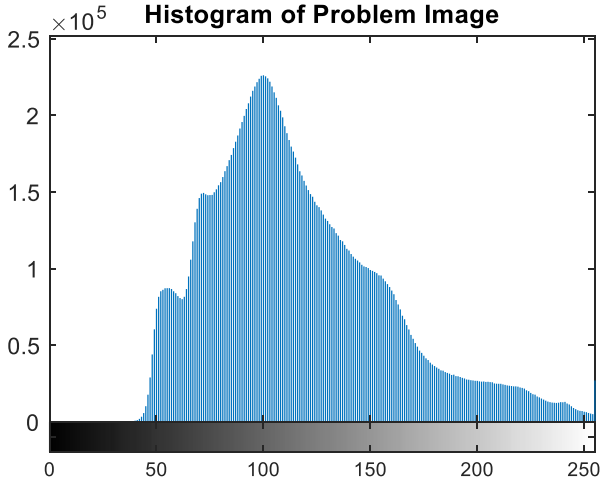
Grey Level	0	1	2	3	4	5	6	7
No. of pixels	0	0	0	790	1023	850	985	448

RGB Image with Color Cast(unwanted tint of color)

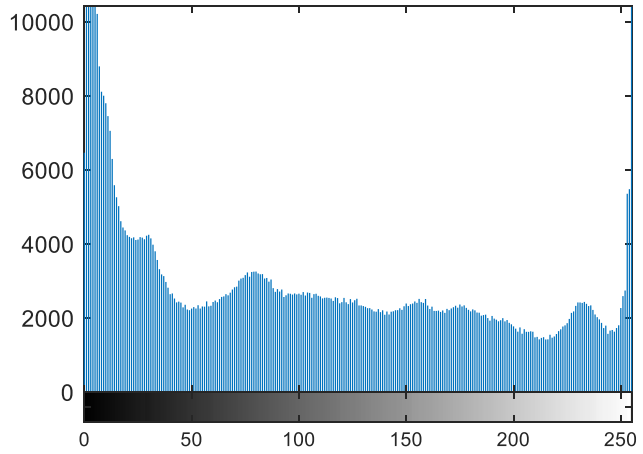
Reference Color Image

Histogram Matched Image

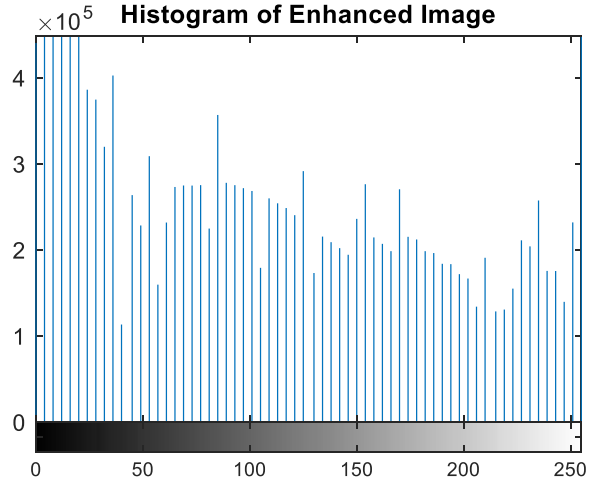
Histogram of Problem Image



Histogram of Reference Image



Histogram of Enhanced Image

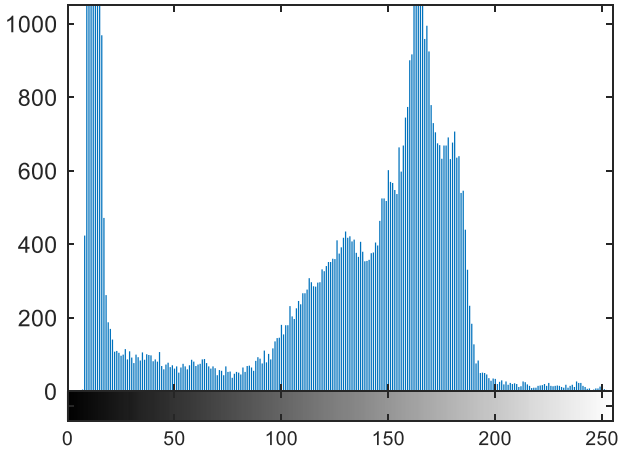


Problem Image

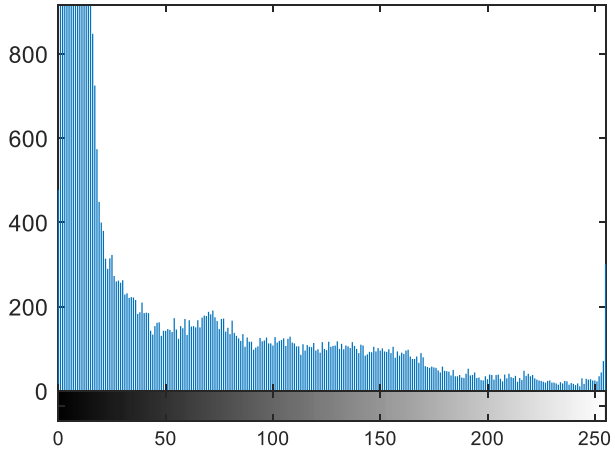
Reference/Target Image

Histogram Matched Image

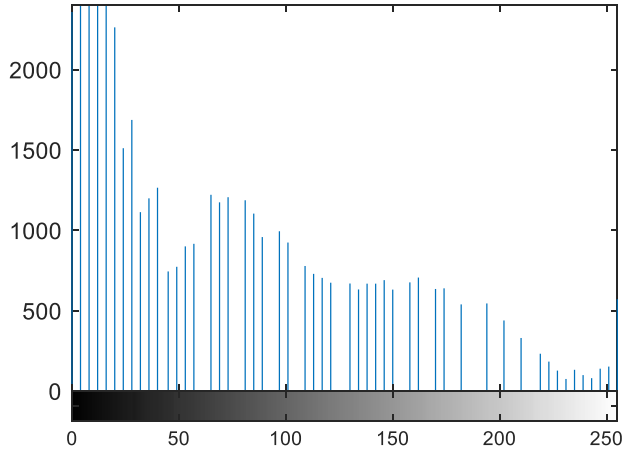
Histogram of Problem Image



Histogram of Reference Image



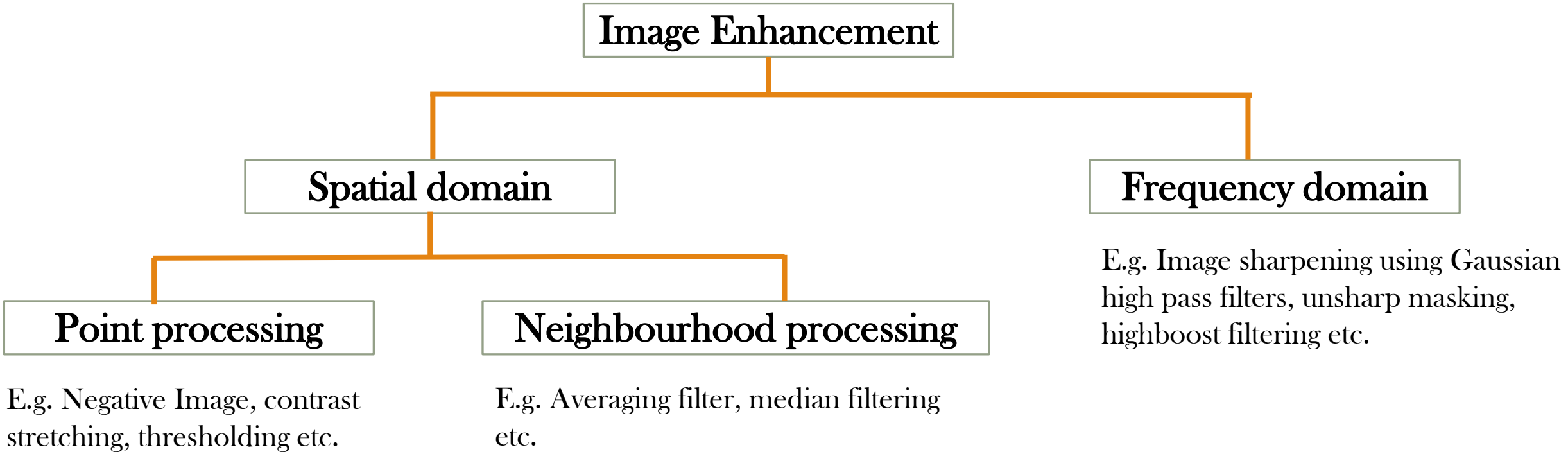
Histogram of Enhanced Image



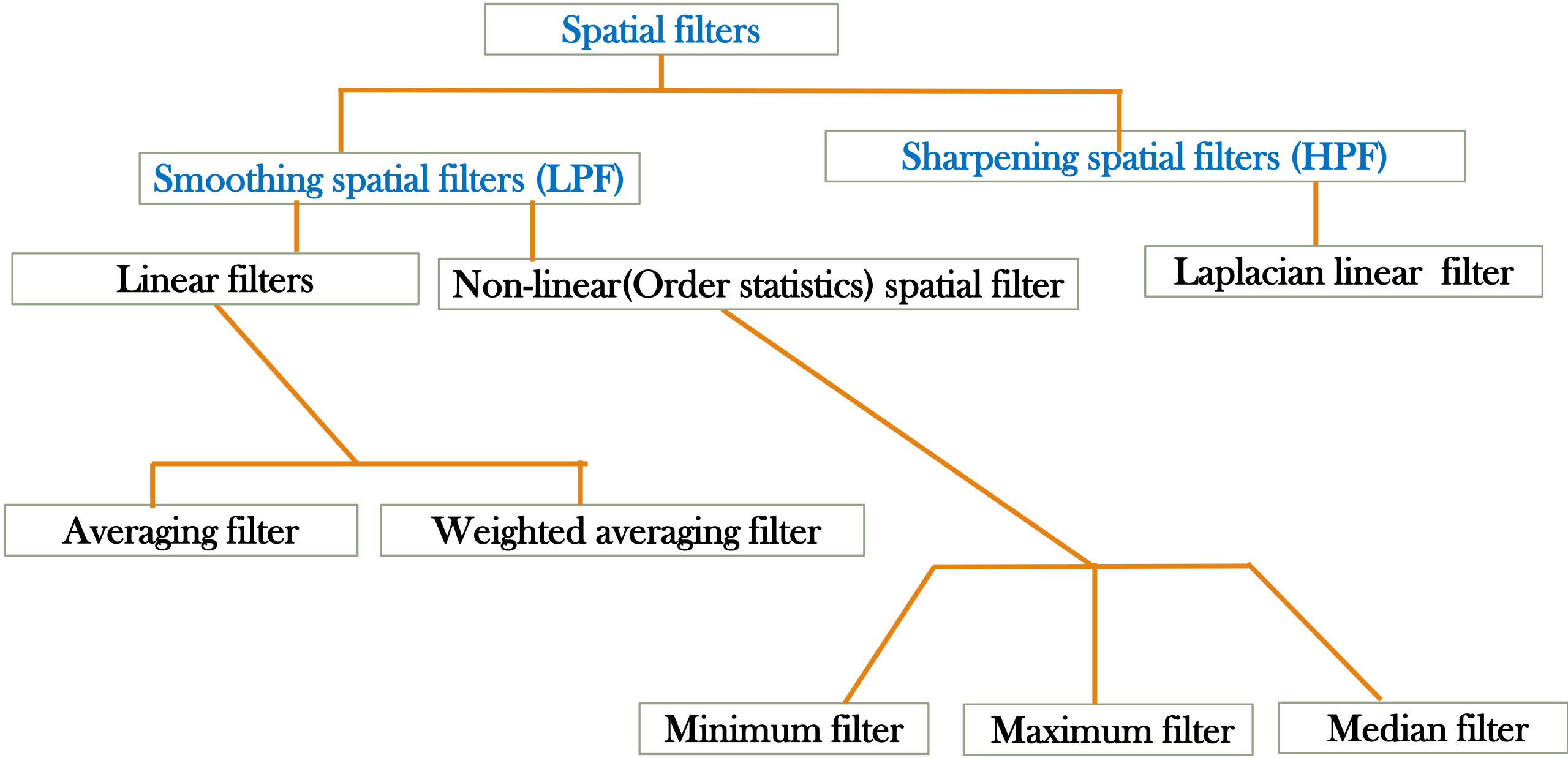
Thank You

15

Fundamentals of Spatial filtering and Smoothing spatial filters in DIP and its implementation in MATLAB

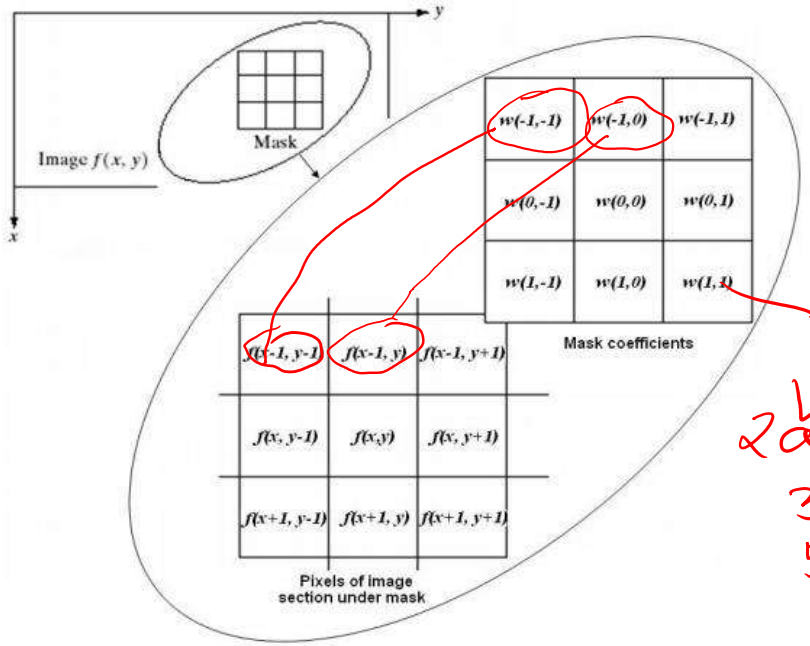


- Spatial filtering change the grey level of a pixel (x,y) depending on the pixel values in a square neighborhood centered at (x,y) using a matrix(filter, mask, kernel/window).
- There are many things that can be achieved by neighborhood processing which are not possible with point processing.


Spatial filters
Smoothing spatial filters (LPF)
Sharpening spatial filters (HPF)
Linear filters
Non-linear(Order statistics) spatial filter
Laplacian linear filter
Averaging filter
Weighted averaging filter
Minimum filter
Maximum filter
Median filter

❖ Averaging Linear Filtering:

- To achieve neighborhood processing, a 3 x 3 mask (or 5 x 5, 7 x 7....) on the image, multiply each component of the mask with the corresponding value of the image, add them up and place the value that we get, at the center. The operation being same as convolution.
- This smoothing process is used for blurring sharp edges.



$$R = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \cdots$$
$$+w(0,0)f(x,y) + \cdots + w(1,0)f(x+1,y) + w(1,1)f(x+1,y+1)$$

$m \times n$

$2a+1 \quad 2b+1$

3×3

$5 \times 5 \ldots$

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t)f(x+s,y+t)$$

where $a = (m-1)/2$ and $b = (n-1)/2$

Filter mask

Box

1	1	1
1	1	1
1	1	1

x 1/9

Weight w

Input image

200	215	212	208	196
198	5	202	199	221
199	207	202	201	211
203	218	210	210	198
203	218	210	0	198
200	215	212	208	205

Output image

181	184	186	206	205
180	182	183	206	207
181	183	184	206	208
206	208	186	182	181
207	210	189	183	180
206	209	189	184	181

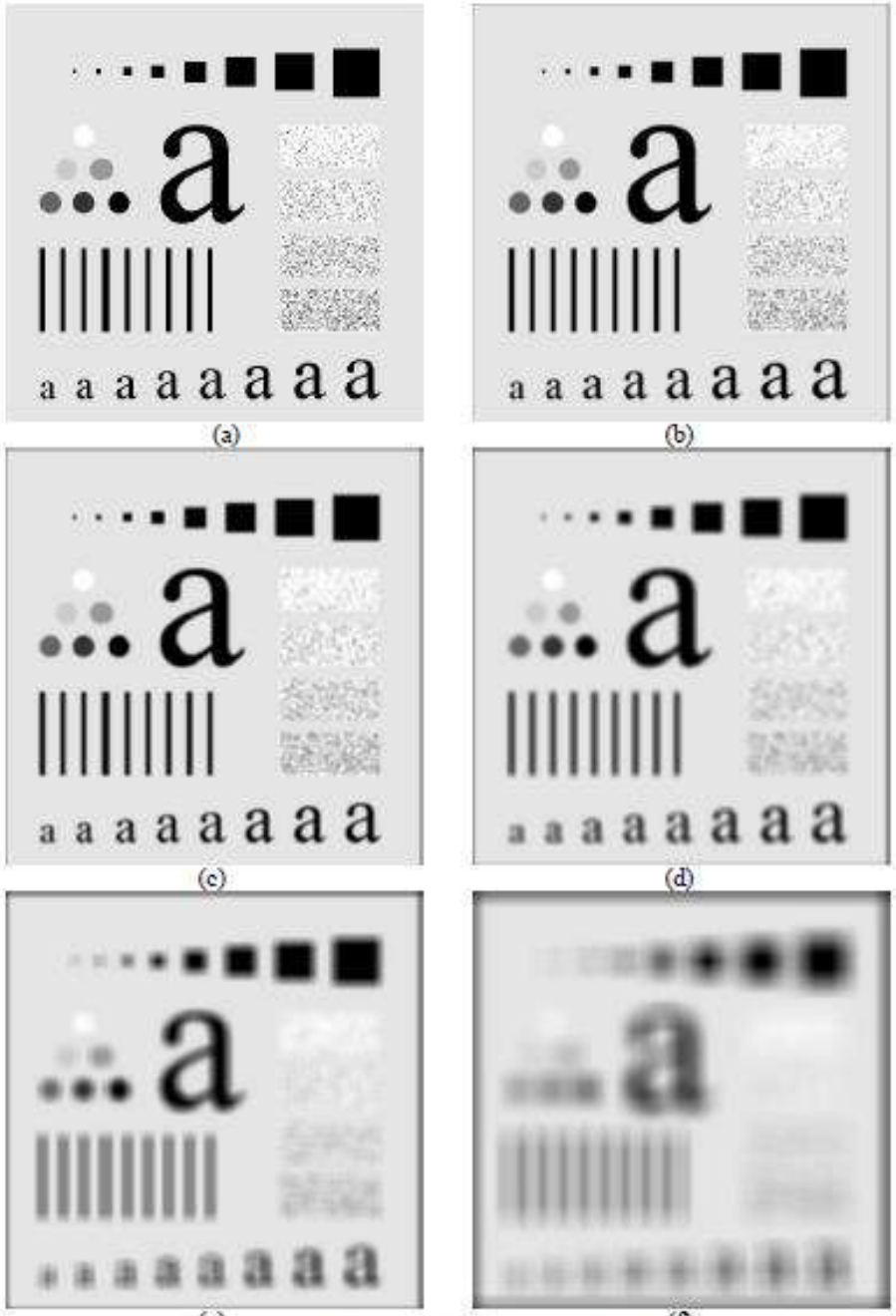
$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \times \frac{1}{16}$$

$(1\times200 + 1\times215 + 1\times212 + 1\times198 + 1\times5 + 1\times202 + 1\times199 + 1\times207 + 1\times202)/9$

$= 1640/9 = 182.22 \rightarrow 182$

Weighted Average Filter

- The two noises are replaced with the average of their surrounding points. The process of reducing the influence of noise is called smoothing or blurring.



Effect of averaging filter. (a) Original image. (b)-(f) Results of smoothing with square averaging filter masks of sizes n = 3,5,9,15, and 35, respectively.

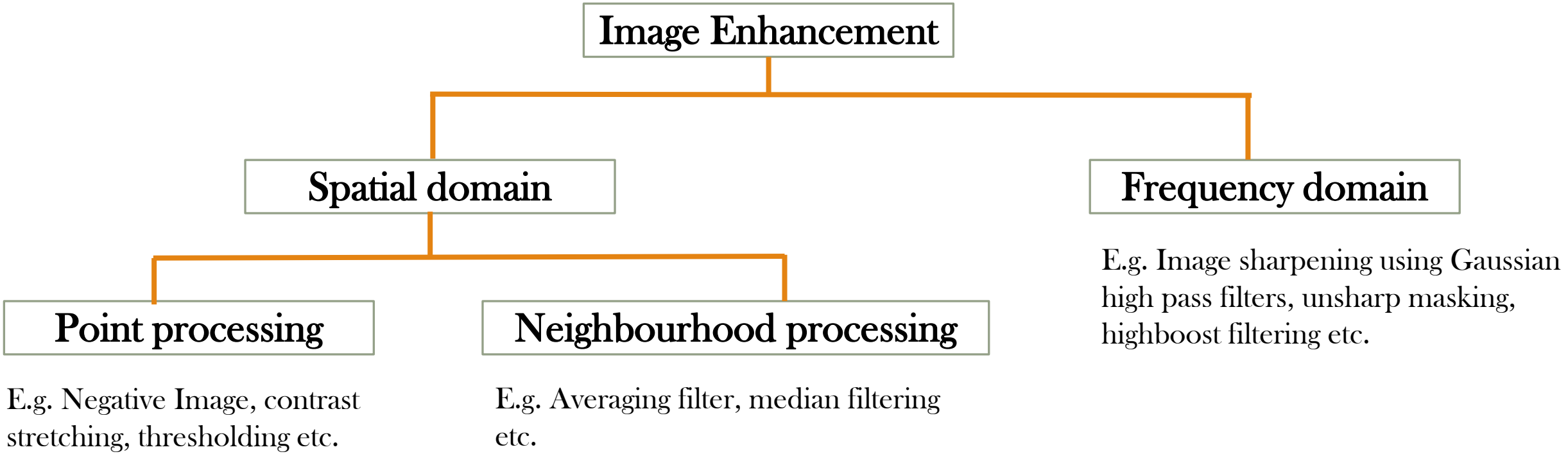
Image with random Gaussian noise

LPF (Smoothed) Image

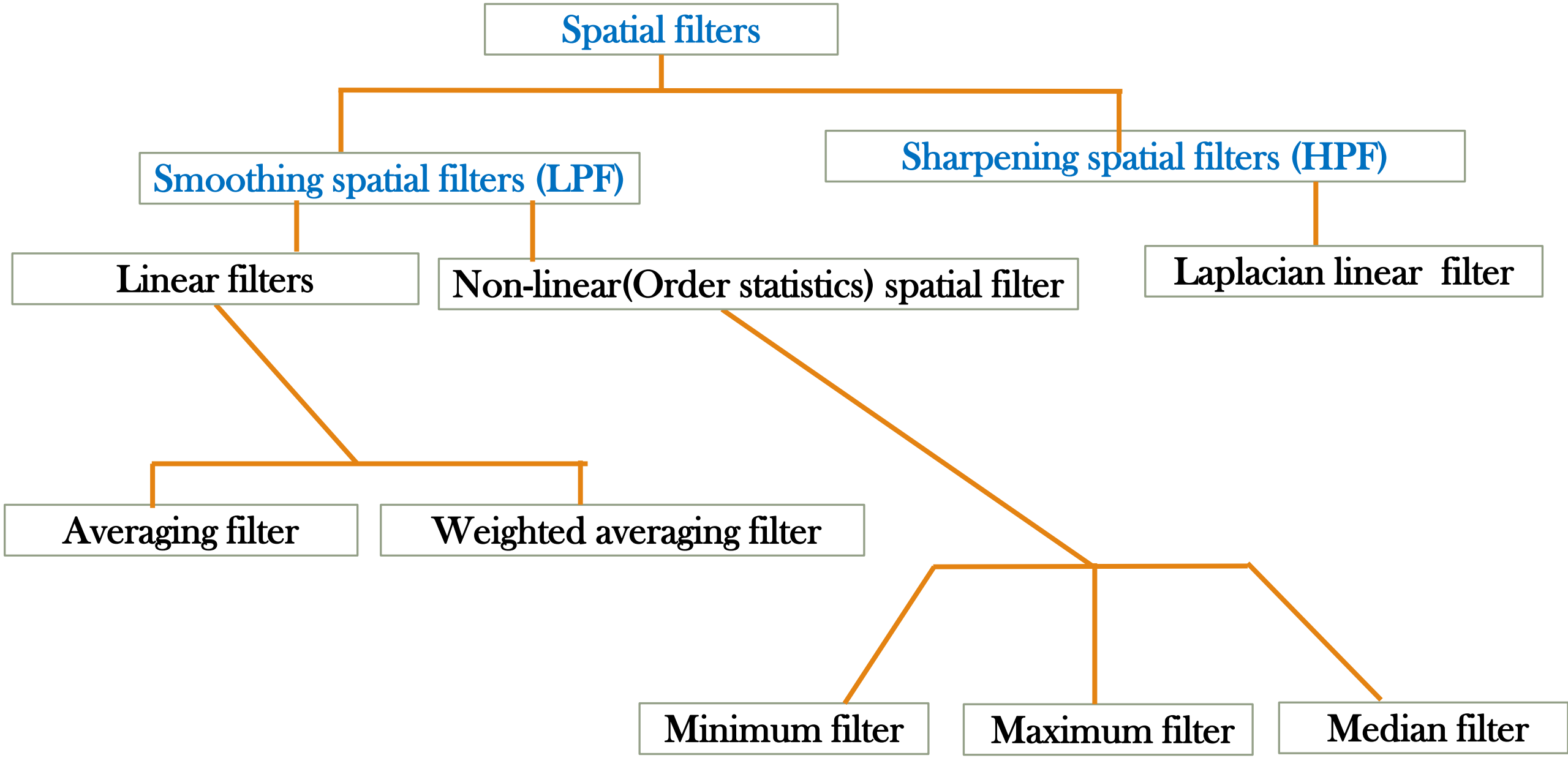
Thank You

16

Order statistics/Non-linear filters in DIP and its implementation in MATLAB



- Spatial filtering change the grey level of a pixel (x,y) depending on the pixel values in a square neighborhood centered at (x,y) using a matrix(filter, mask, kernel/window).
- There are many things that can be achieved by neighborhood processing which are not possible with point processing.

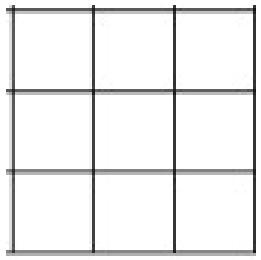

Spatial filters
Smoothing spatial filters (LPF)
Sharpening spatial filters (HPF)
Linear filters
Non-linear(Order statistics) spatial filter
Laplacian linear filter
Averaging filter
Weighted averaging filter
Minimum filter
Maximum filter
Median filter

❖ Non-Linear Order statistics Spatial Filtering:

- Median filtering is a nonlinear method used to remove noise from images.
- It is widely used as it is very effective at removing noise while preserving edges.
- It is particularly effective at removing 'salt and pepper' type noise.
- The median filter works by moving through the image pixel by pixel, replacing each value with the median value of neighboring pixels.

$R = \text{median}\{Z_k \mid k = 1,2,3,.....9\}$

Filter mask

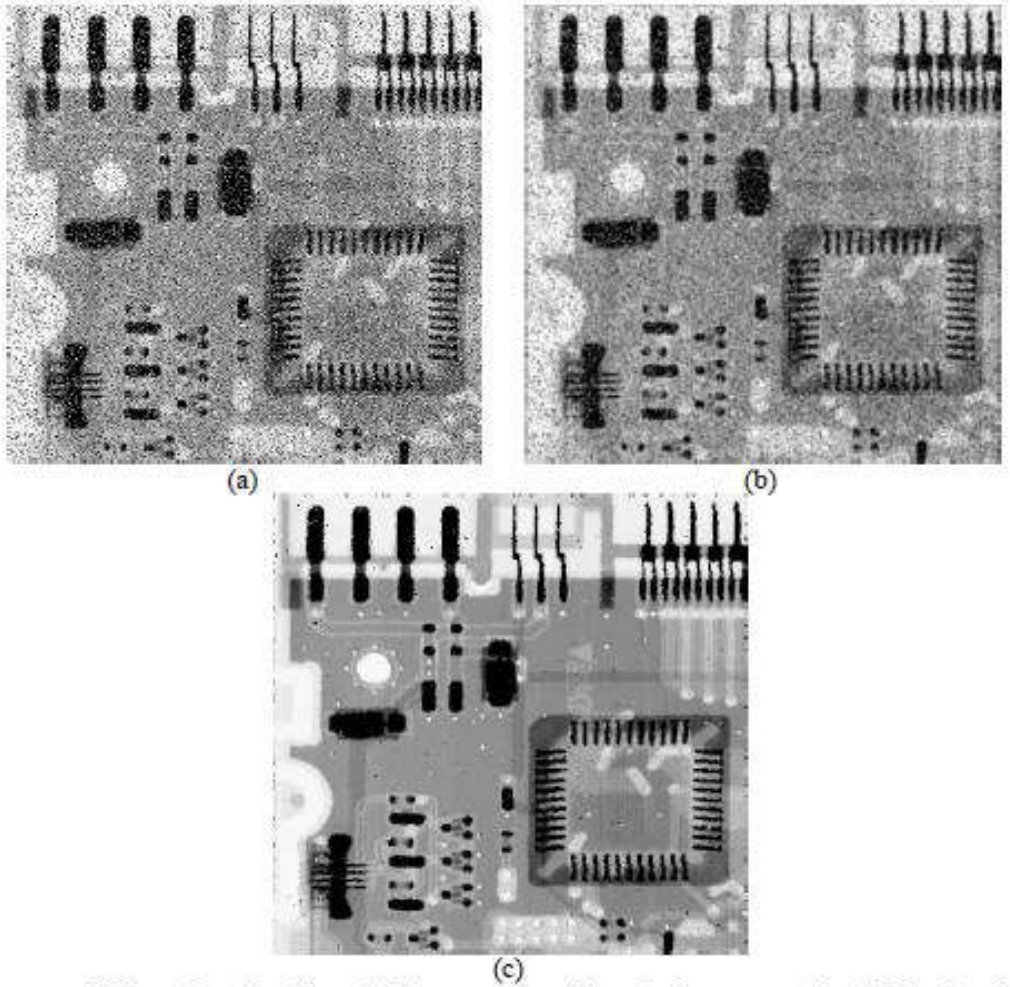


Input image

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	250	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	250	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

Output image

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50
50	50	50	50	50	50	50	50

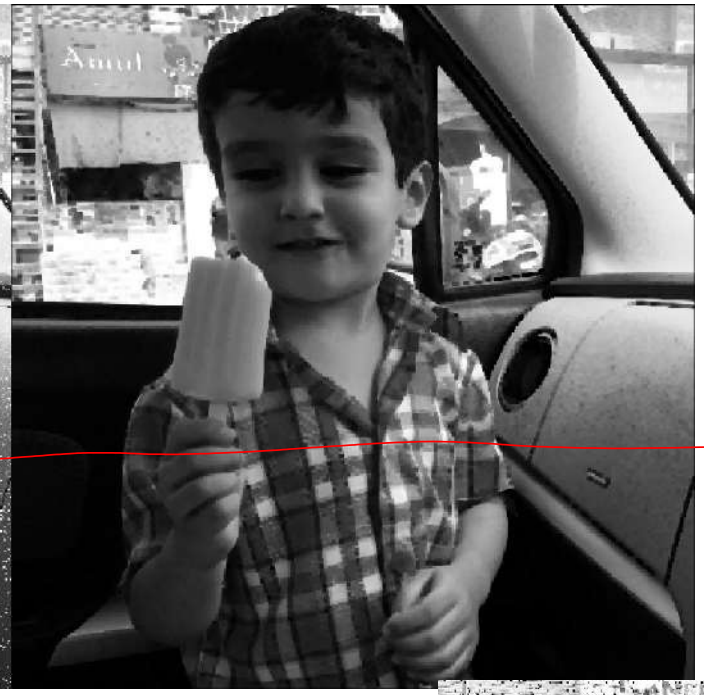


Effect of median filter. (a) Image corrupted by salt & pepper noise. (b) Result of applying 3×3 standard averaging filter on (a). (c) Result of applying 3×3 median filter on (a).

- The effects of median filter are: (1) Noise reduction,
  (2) Less blurring than averaging filter

Image with Salt noise

Minimum filtered Image



Minimum filter:

$R = \min\{Z_k \mid k = 1,2,3,.....9\}$

It is used for finding the darkest point.

It is used for removing the salt noise.

255 → White

0 → Black

Maximum filter:

$R = \max\{Z_k \mid k = 1,2,3,.....9\}$

It is used for finding the brightest point.

It is used for removing the pepper noise.

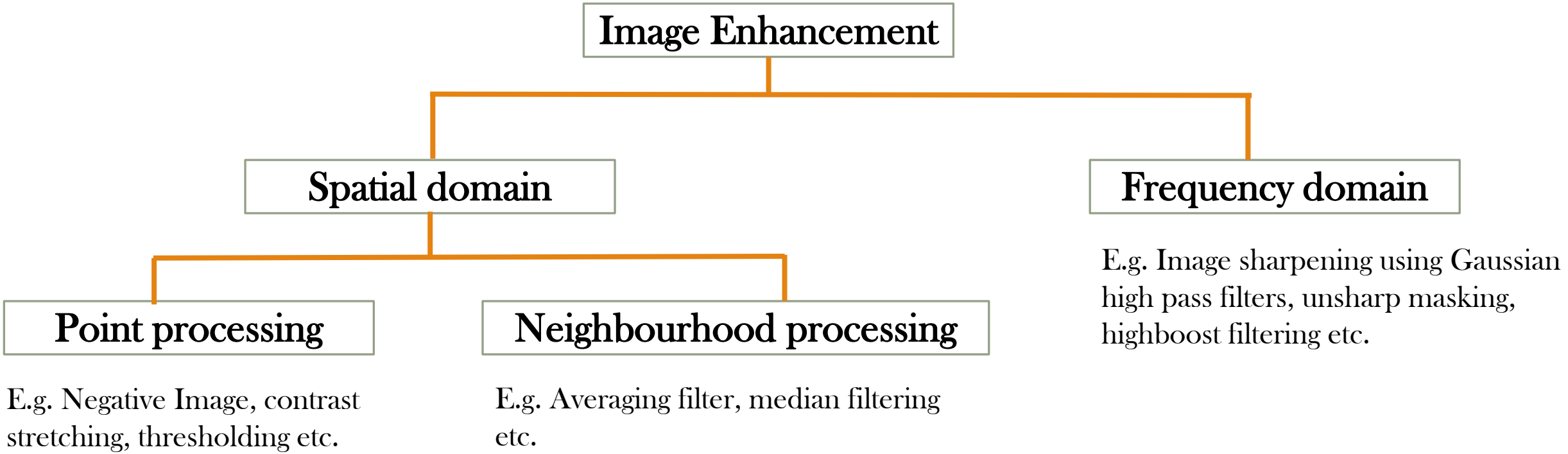
Image with Pepper noise

Maximum filtered Image

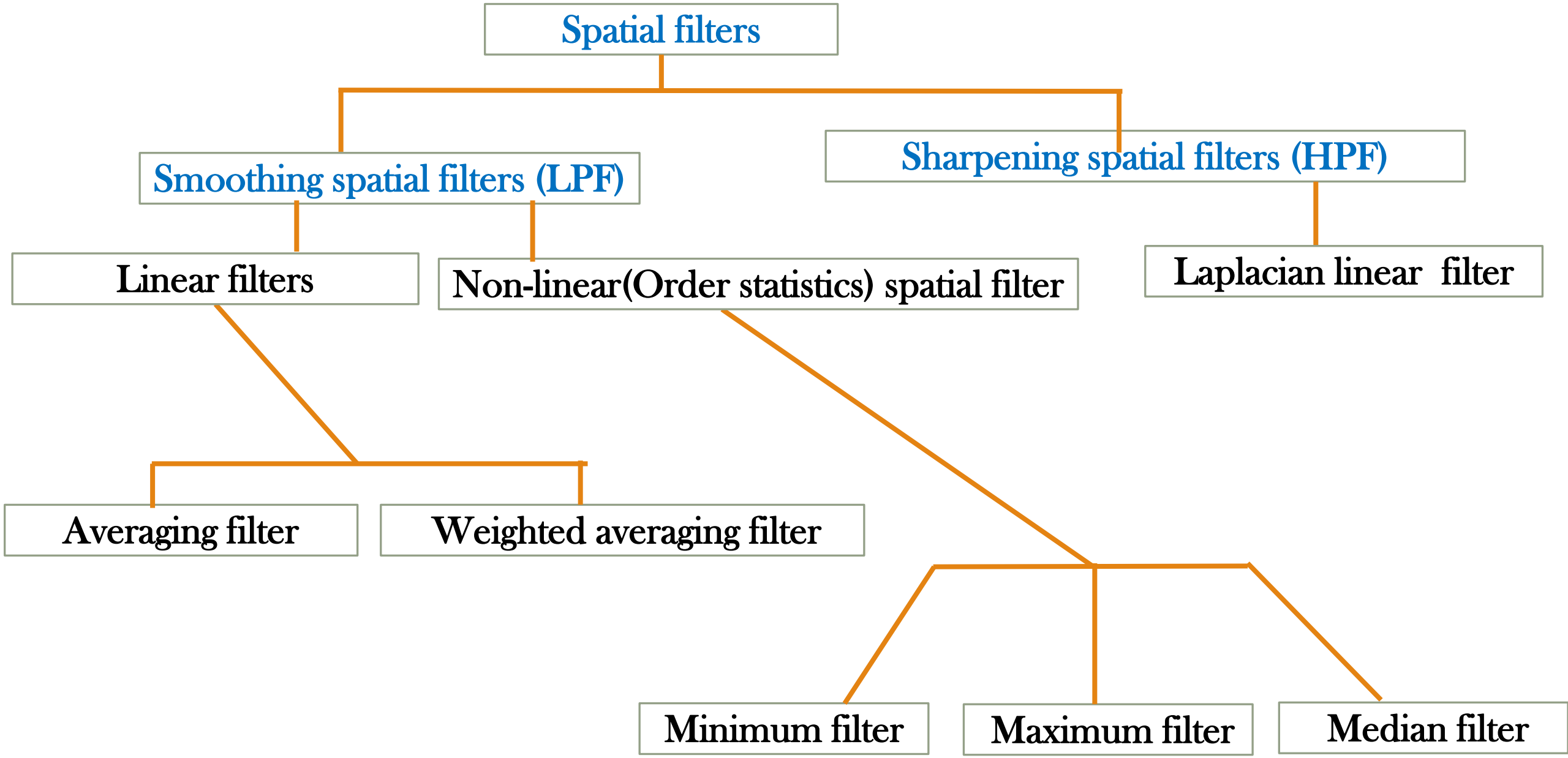
Thank You

17

Image Sharpening spatial filters in DIP and its implementation in MATLAB



- Spatial filtering change the grey level of a pixel (x,y) depending on the pixel values in a square neighborhood centered at (x,y) using a matrix(filter, mask, kernel/window).
- There are many things that can be achieved by neighborhood processing which are not possible with point processing.


Spatial filters
Smoothing spatial filters (LPF)
Sharpening spatial filters (HPF)
Linear filters
Non-linear(Order statistics) spatial filter
Laplacian linear filter
Averaging filter
Weighted averaging filter
Minimum filter
Maximum filter
Median filter

❖ Sharpening Spatial Filters:

- Sharpening filters are used to highlight fine details (e.g. edges) in an image, or enhance details that are blurred through errors or imperfect capturing devices.
- Sharpening is opposite of smoothing(averaging). Hence in mathematics, it is given by partial derivatives(HPF) as averaging is integration(LPF).
- While linear smoothing is based on the weighted summation or integral operation on the neighborhood, the sharpening is based on the derivative (gradient) or finite difference.
- In smoothing we try to smooth noise and ignore edges and in sharpening we try to enhance edges and ignore noise.

❖ Partial derivatives of digital image:

- The **first order** partial derivative of the digital image f(x,y) are:

$$\frac{\partial f}{\partial x} = f(x+1,y) - f(x,y) \quad \text{and} \quad \frac{\partial f}{\partial y} = f(x,y+1) - f(x,y)$$

- The first derivative must be:

(1) Zero along flat segments (i.e. constant grey levels)
(2) Non-zero at the outset of grey level step or ramp (edges or noise)
(3) Non-zero along segments of continuing changes (i.e. ramps)

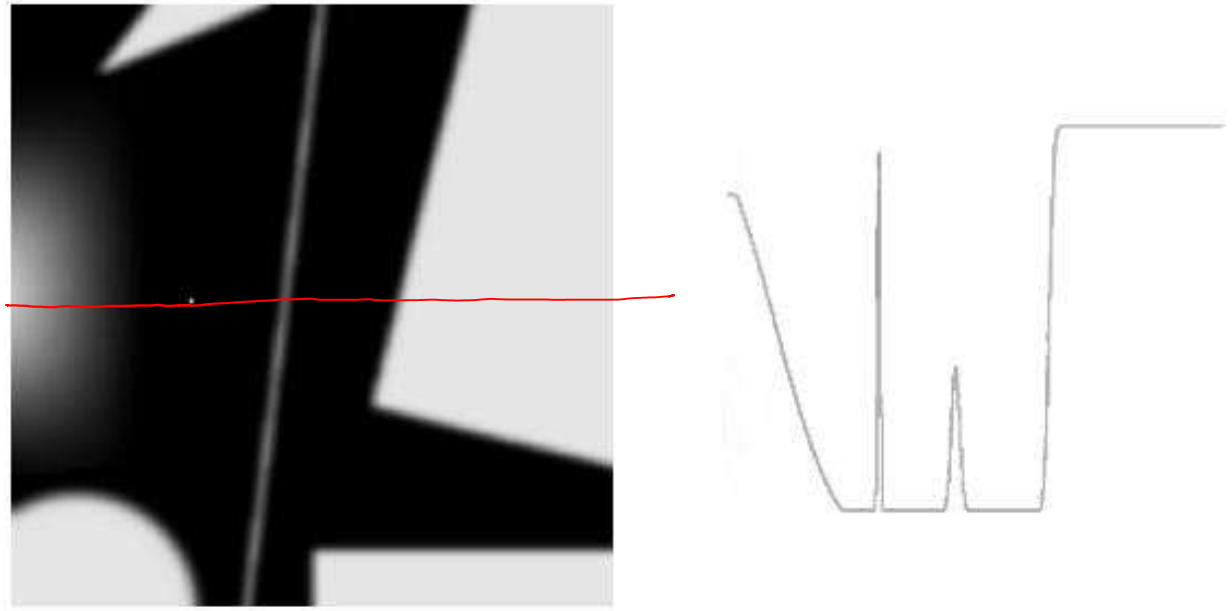
- The **second order** partial derivatives of the digital image f(x,y) are:

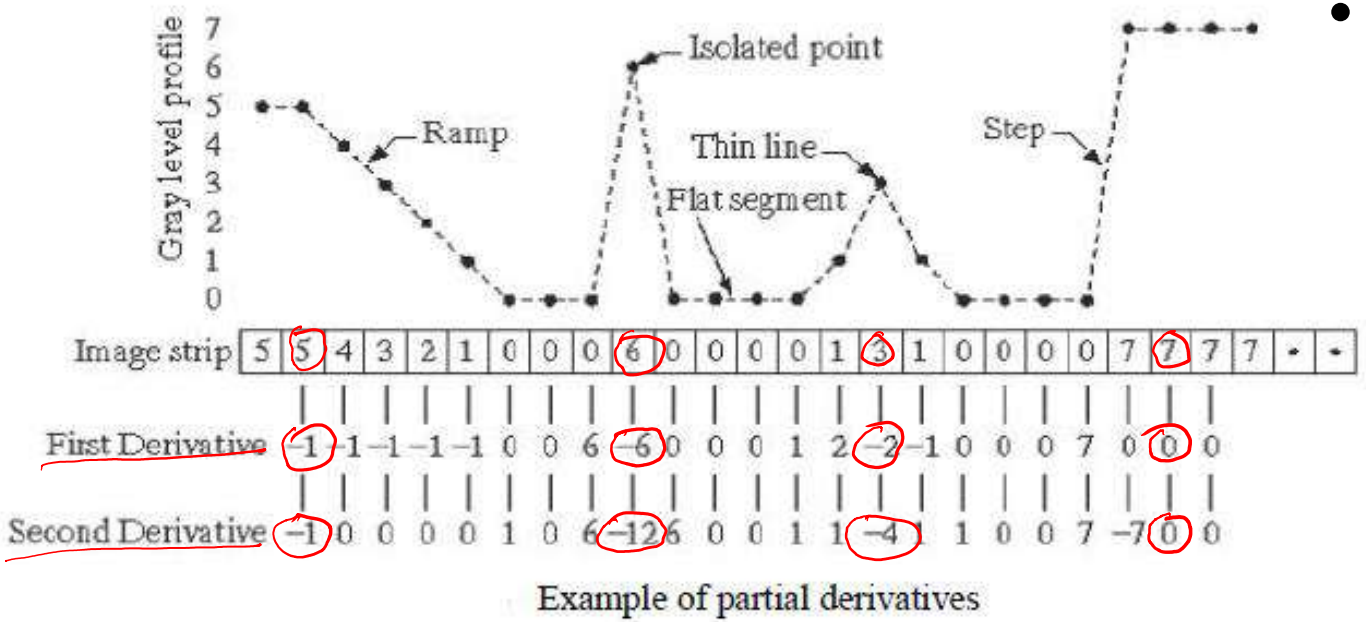
$$\frac{\partial^2 f}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

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

- The second derivative must be:

(1) Zero along flat segments (i.e. constant grey levels)
(2) Non-zero at the outset of grey level step or ramp (edges or noise)
(3) Zero along ramps.





Example of partial derivatives

- The first derivative detects thick edges while second derivative detects thin edges. $f' = f(x+1) - f(x)$
- Second derivative has much stronger response at grey level step than first derivative. $f'' = f(x+1) + f(x-1) - 2f(x)$
- Thus we can expect a second order derivative to enhance fine detail (thin lines, edges, including noise) much more than a first order derivative.

❖ The Laplacian filter:

- The Laplacian operator of an image f(x,y) is:

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

$$\nabla^2 f(x, y) = f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1) - 4f(x, y)$$

0	1	0
1	-4	1
0	1	0

- This equation can be implemented using the 3 x 3 mask:

- As the Laplacian filter is a linear spatial filter, we can apply it using the same mechanism of convolution. This will produce a Laplacian image that has grayish edge lines and other discontinuities, all superimposed on a dark, featureless background.
- Background features can be "recovered" while still preserving the sharpening effects of the Laplacian simply by adding the original and Laplacian images.

$$g(x, y) = f(x, y) + \nabla^2 f(x, y)$$

Filter mask

0	1	0
1	-4	1
0	1	0

Input image

20	20	20	20	20
20	5	20	20	20
20	20	20	20	20
20	20	20	5	20
20	20	20	20	20

Output image

0	-15	0	0	0
0	60	0	0	0
0	-15	0	-15	0
0	0	0	60	0
0	0	0	-15	0

$= 0+20+0+20-20+20+0+20+0 = 60.$

$= 0+20+0+20-80+20+0+5-10 = -15$

(a)

(b)

(c)

Example of applying Laplacian filter. (a) Original image. (b) Laplacian image. (c) Sharpened image.

Original Image

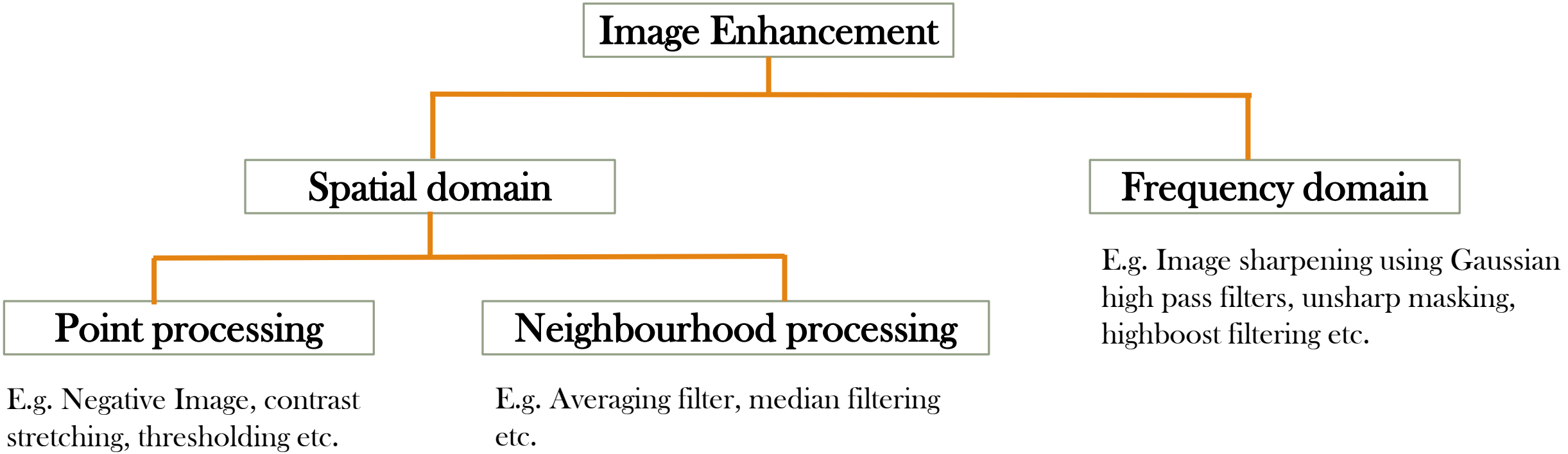
Laplacian filtered Image

Sharpened Image (Laplacian+Original)

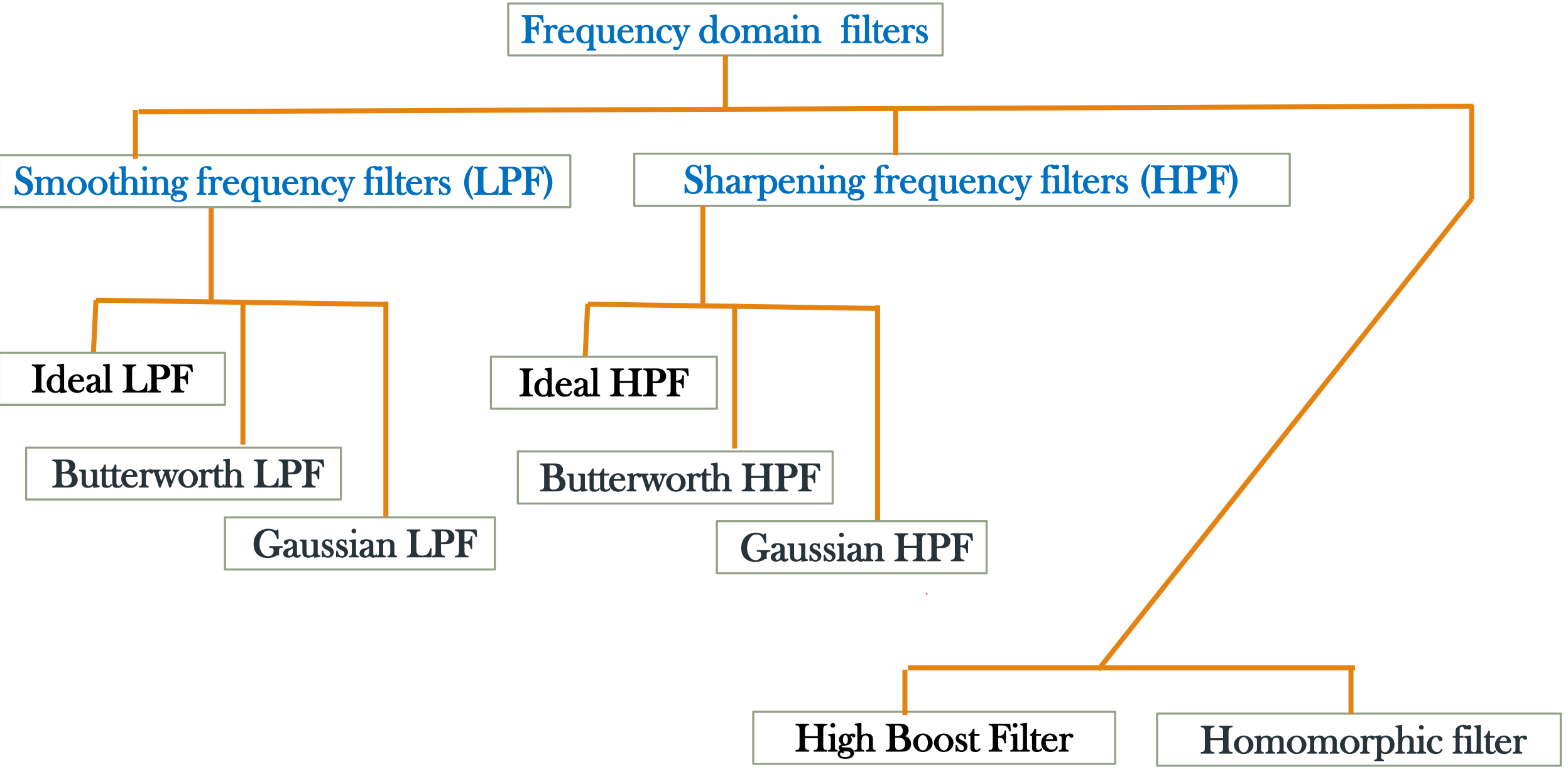
Thank You

18

Introduction to Image Enhancement in the frequency domain and different steps for filtering in the frequency domain



- The frequency content of an image refers to the rate at which the gray levels change in time. Rapidly changing brightness values correspond to high frequency terms, slowly changing brightness values correspond to low frequency terms. The Fourier transform is a mathematical tool that analyses a signal (e.g. images) into its spectral components depending on its frequency content.


Frequency domain filters
Smoothing frequency filters (LPF)
Sharpening frequency filters (HPF)
Ideal LPF
Butterworth LPF
Gaussian LPF
Ideal HPF
Butterworth HPF
Gaussian HPF
High Boost Filter
Homomorphic filter

Frequency domain vs. Spatial domain

	Frequency domain	Spatial domain
1.	is resulted from Fourier transform	is resulted from sampling and quantization
2.	refers to the space defined by values of the Fourier transform and its frequency variables (u,v).	refers to the image plane itself, i.e. the total number of pixels composing an image, each has spatial coordinates (x,y)
3.	has complex quantities	has integer quantities

$F(u,v) \rightarrow \boxed{H(u,v)} \rightarrow G(u,v)$

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

$g(x,y) = F^{-1}(G(u,v))$ multiplication

$f(x,y) \rightarrow \boxed{h(x,y)} \rightarrow g(x,y)$

$g(x,y) = f(x,y) * h(x,y)$

❖ Advantages of filtering in frequency domain:

- The frequency domain filtering is advantageous because it is computationally faster to perform two 2D Fourier transforms and a filter multiply than to perform a convolution in the spatial domain.

- Frequency domain gives you control over the whole images, where we can enhance(eg. Edges) and suppress(eg. Smooth shadow) different characteristics of the image very easily.

- Frequency domain has a established suit of processes and tools that can be borrowed directly from signal processing in other domain.

- Image enhancement in the frequency domain is straightforward. The idea of blurring an image by reducing its high frequency components, or sharpening an image by increasing the magnitude of its high frequency components is intuitively easy to understand.

- Fourier transform of a discrete function of one variable $f(x), x = 0, 1, 2, \ldots, M-1$ (discrete Fourier transform, or DFT)

$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M} \qquad \text{for } u = 0, 1, 2, \ldots, M-1$$

- Inverse DFT

$$f(x) = \sum_{u=0}^{M-1} F(u) e^{j2\pi ux/M} \qquad \text{for } x = 0, 1, 2, \ldots, M-1$$

  - Two-dimensional DFT and its inverse
    - In 2D, DFT of an image $f(x, y)$ of dimensions $M \times N$ is given by

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}$$

    - The inverse Fourier transform is given by

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)}$$

The domain of $\boldsymbol{u}$ and $\boldsymbol{v}$ values $\boldsymbol{u} = 0, 1, ..., M\text{-}1$, $\boldsymbol{v} = 0,1,\ldots, N\text{-}1$ is called the frequency domain of $f(x,y)$.

$$R(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\cos\left(2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)\right)$$ is called real part

$$I(u,v) = \frac{-1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\sin\left(2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)\right)$$ is called imaginary part

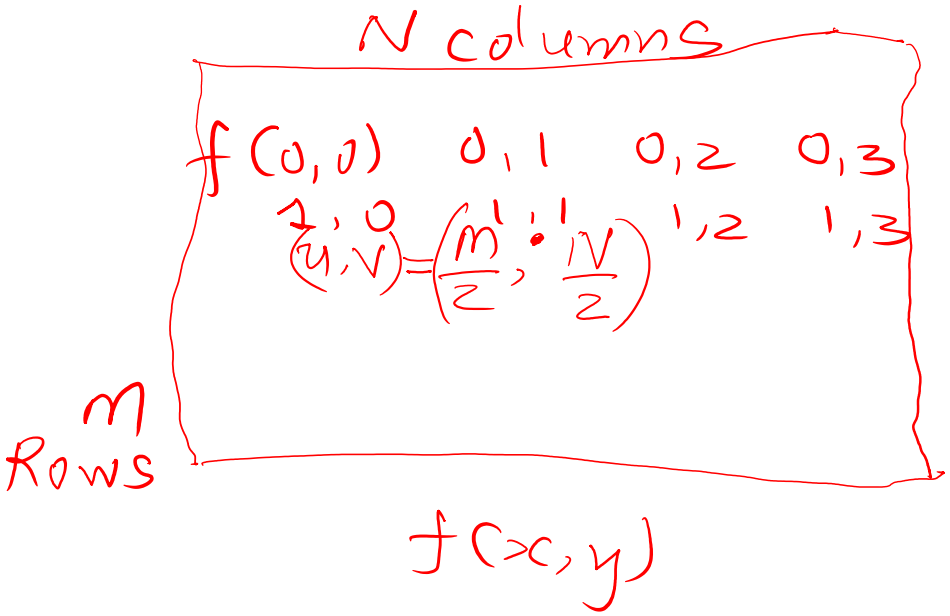
The magnitude of $F(u,v)$, $|F(u,v)|=[R^2(u,v)+I^2(u,v)]^{1/2}$, is called the **Fourier spectrum** of the transform.

The **phase angle** (**phase spectrum**) of the transform is:

$$\emptyset(u,v) = tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$$

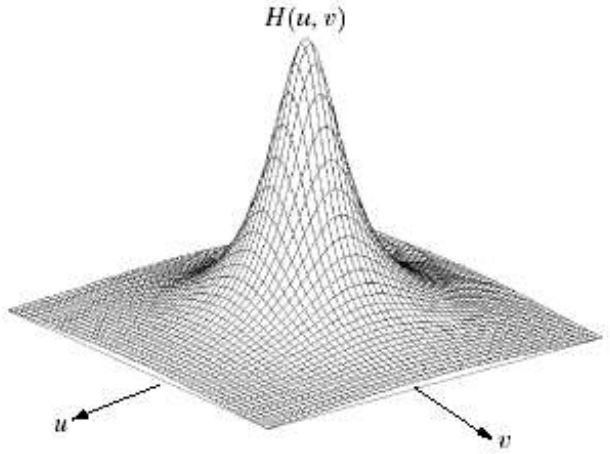
Note that, $F(0,0)$ = the average value of $f(x,y)$ and is referred to as the *dc component* of the spectrum.

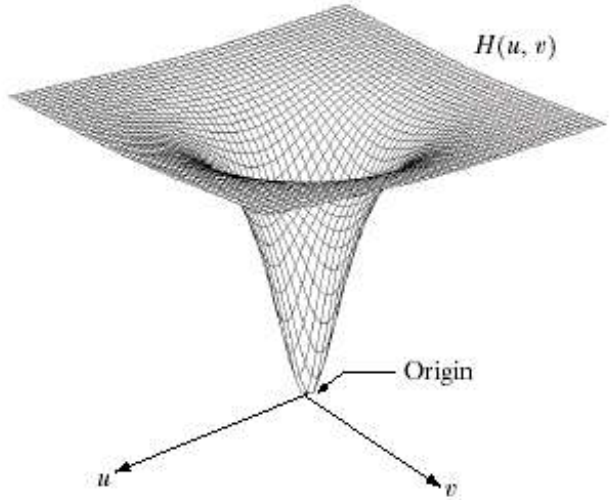
It is a common practice to multiply the image f(x,y) by $(-1)^{x+y}$. In this case, the DFT of $(f(x,y)(-1)^{x+y})$ has its origin located at the centre of the image, i.e. at (u,v) = $(M/2,N/2)$.



$(-1)^{0+0} = (-1)^0 = 1$

Lowpass and Highpass Filters



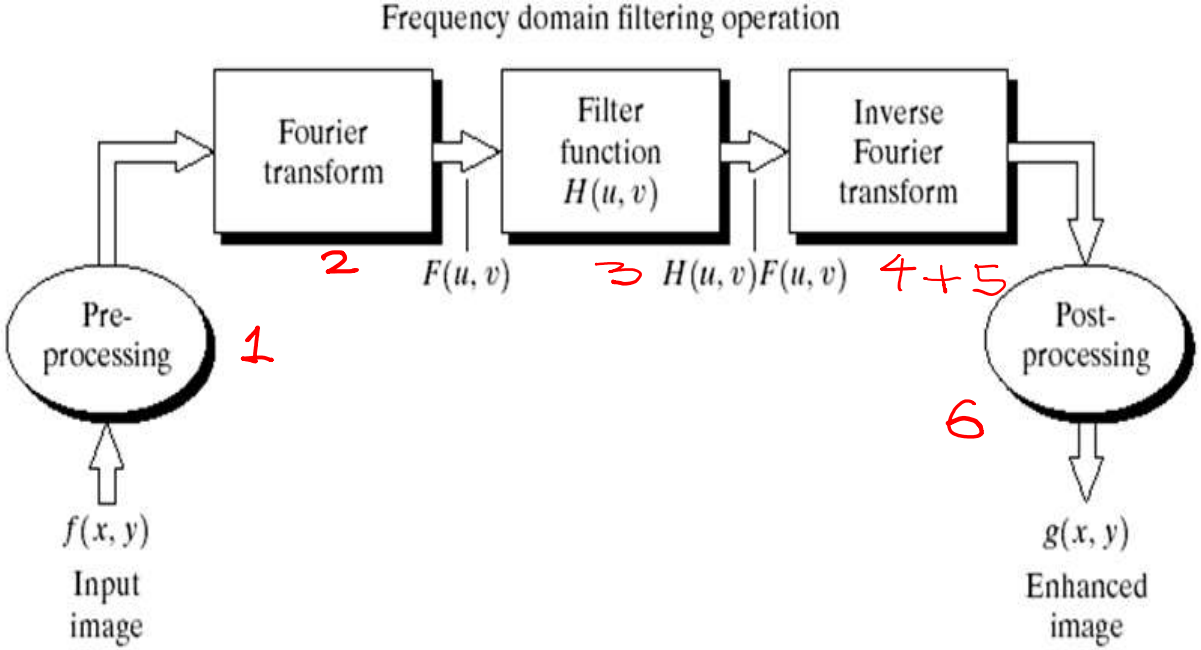


→ Smoothing

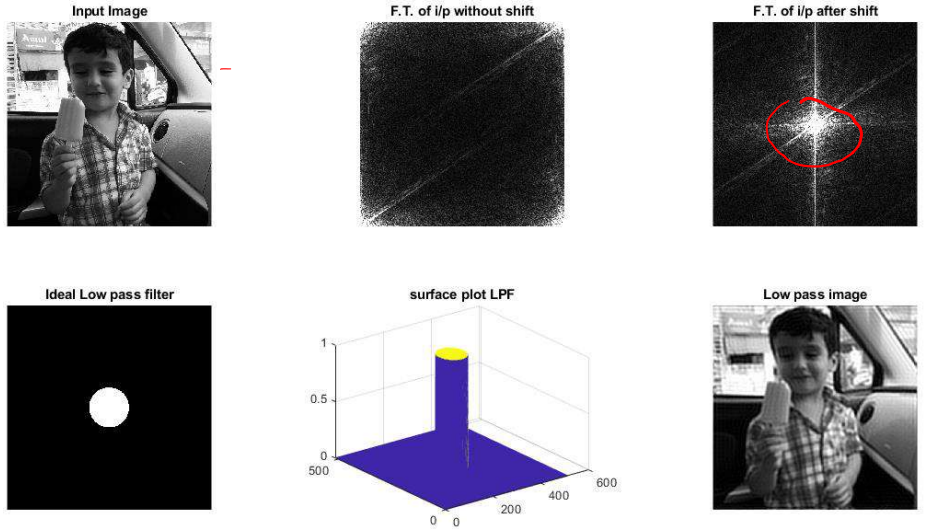
→ Sharpening

❖ Filtering in the Frequency Domain

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



Basic steps for filtering in the frequency domain.



Thank You

19

Image Smoothing in frequency domain filtering and its Implementation in MATLAB

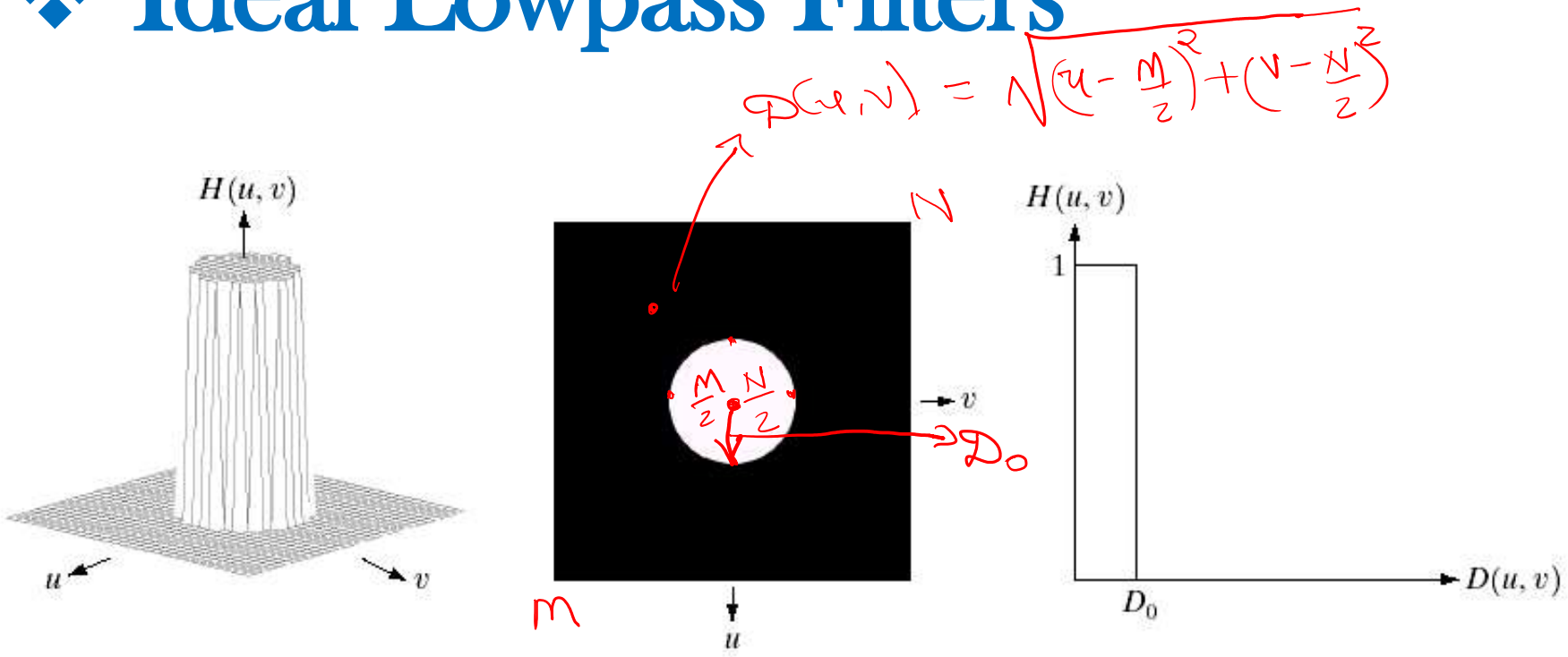
❖ Smoothing Filters

$f(x,y) * h(x,y) = g(x,y) \Rightarrow$ F.T. $\Rightarrow F(u,v) \times H(u,v) = G(u,v)$

$\Leftarrow$ I.F.T. $\Leftarrow$

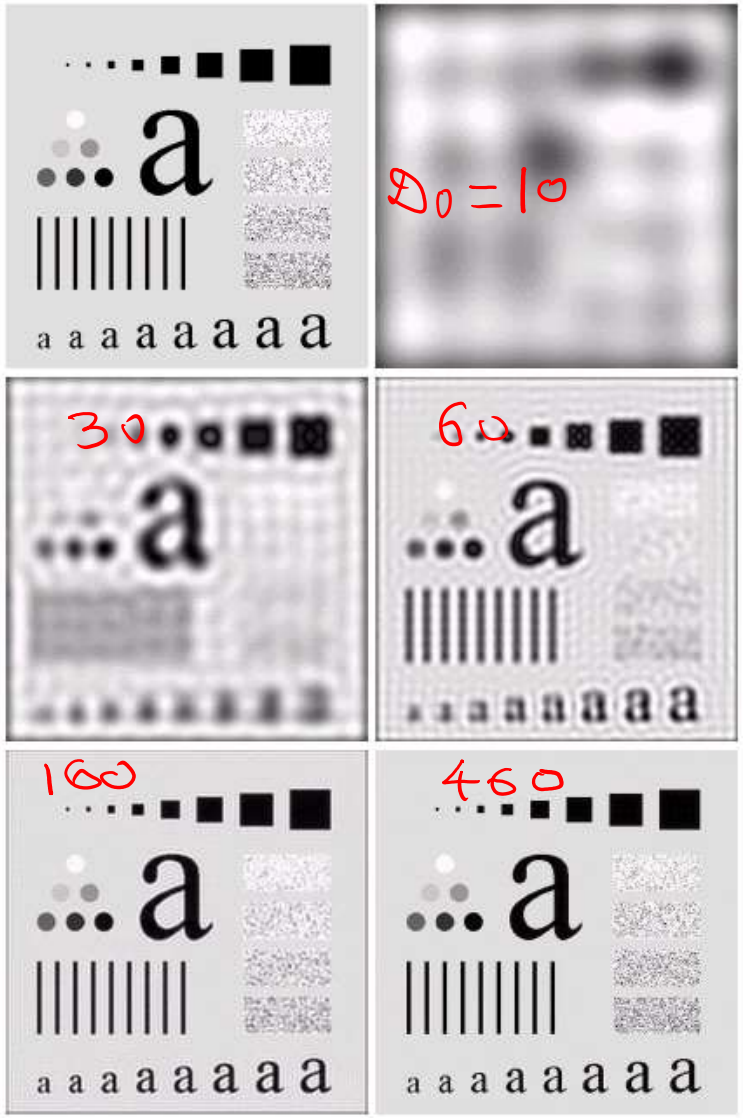
Ideal lowpass filters	$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{otherwise} \end{cases}$	This has a sharp discontinuity and hence, Ringing effect is present
Butterworth lowpass filters	$H(u,v) = \dfrac{1}{1 + [D(u,v)/D_0]^{2n}}$	Does not have a sharp discontinuity and hence ringing is only observed for filters of higher orders.
Gaussian lowpass filters	$H(u,v) = e^{-D^2(u,v)/2D_0^2}$	The profile of GLPF is not as tight as that of BLPF and hence no ringing effect is present.

❖ Ideal Lowpass Filters

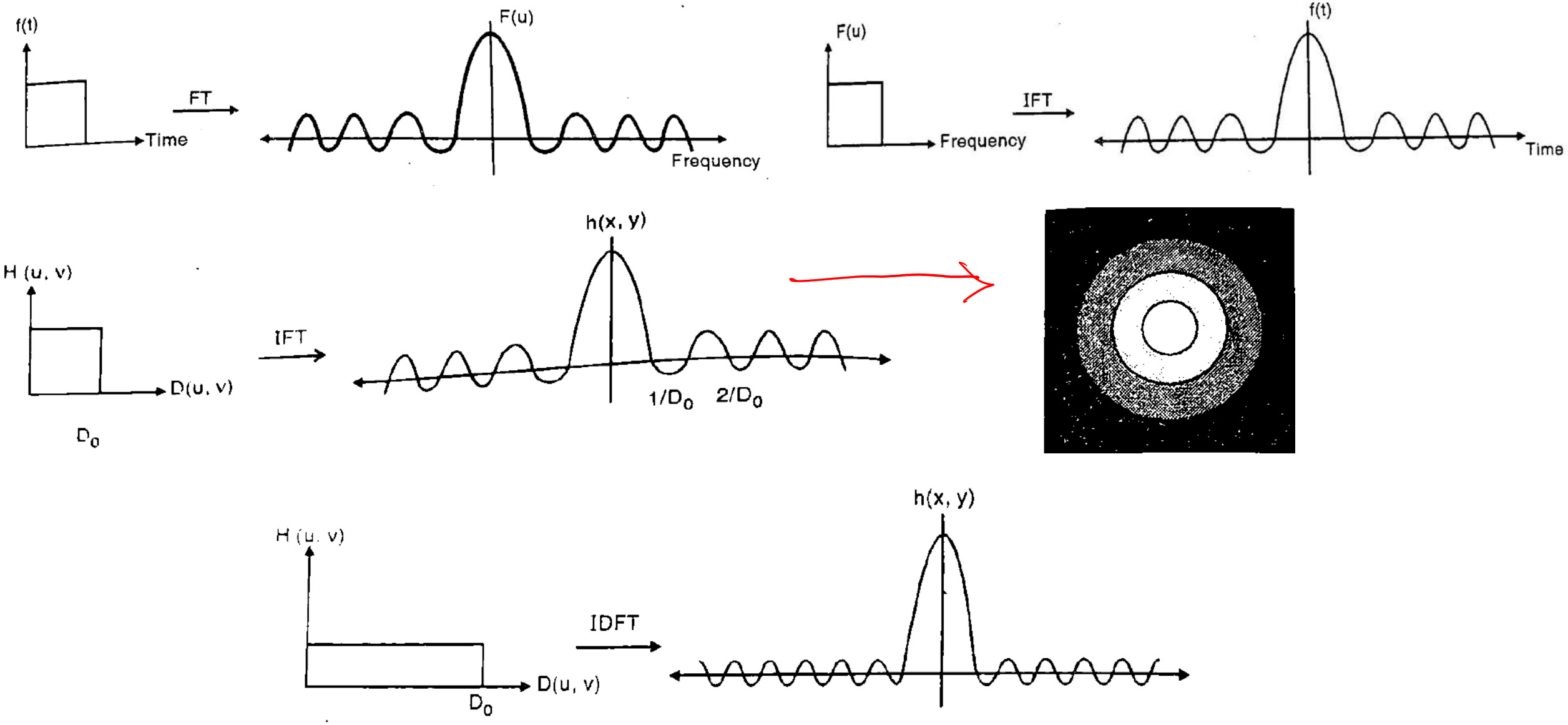


a b c

FIGURE 4.10 (a) Perspective plot of an ideal lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.



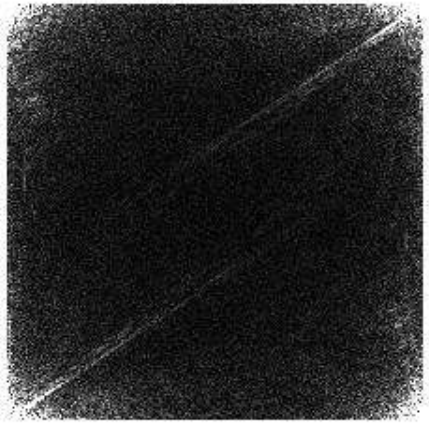
Ringing Effect



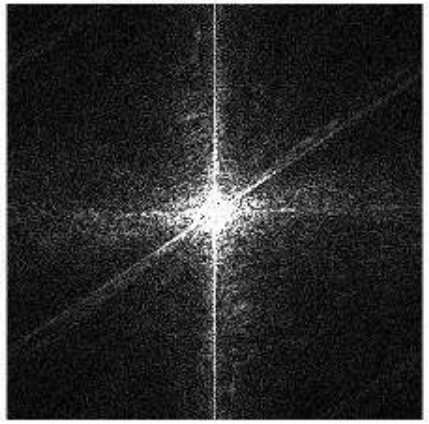
Ideal Lowpass Filters

Input Image

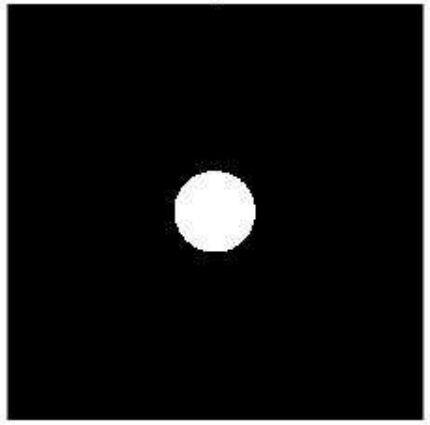
F.T. of i/p without shift



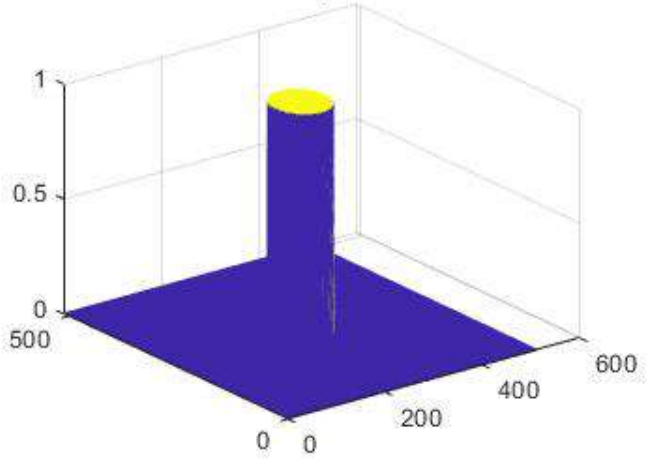
F.T. of i/p after shift



Ideal Low pass filter



surface plot LPF

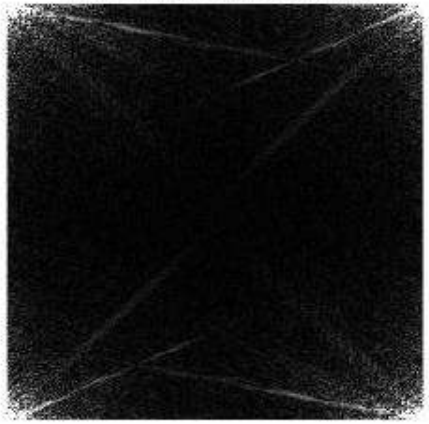


Low pass image

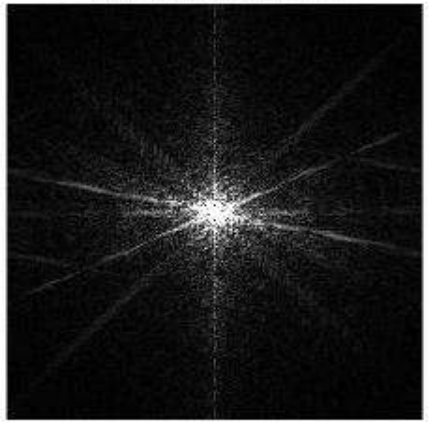
Ideal Lowpass Filters

Input Image

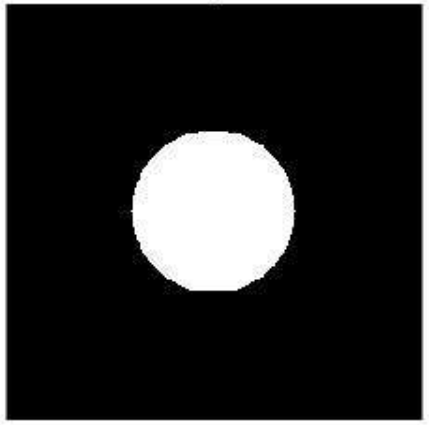
F.T. of i/p without shift



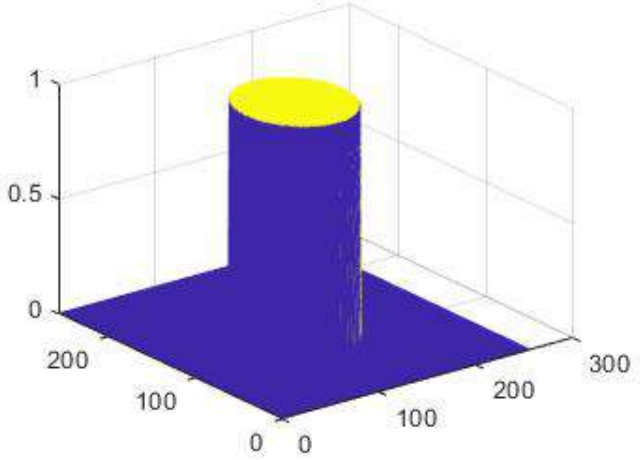
F.T. of i/p after shift



Ideal Low pass filter



surface plot LPF

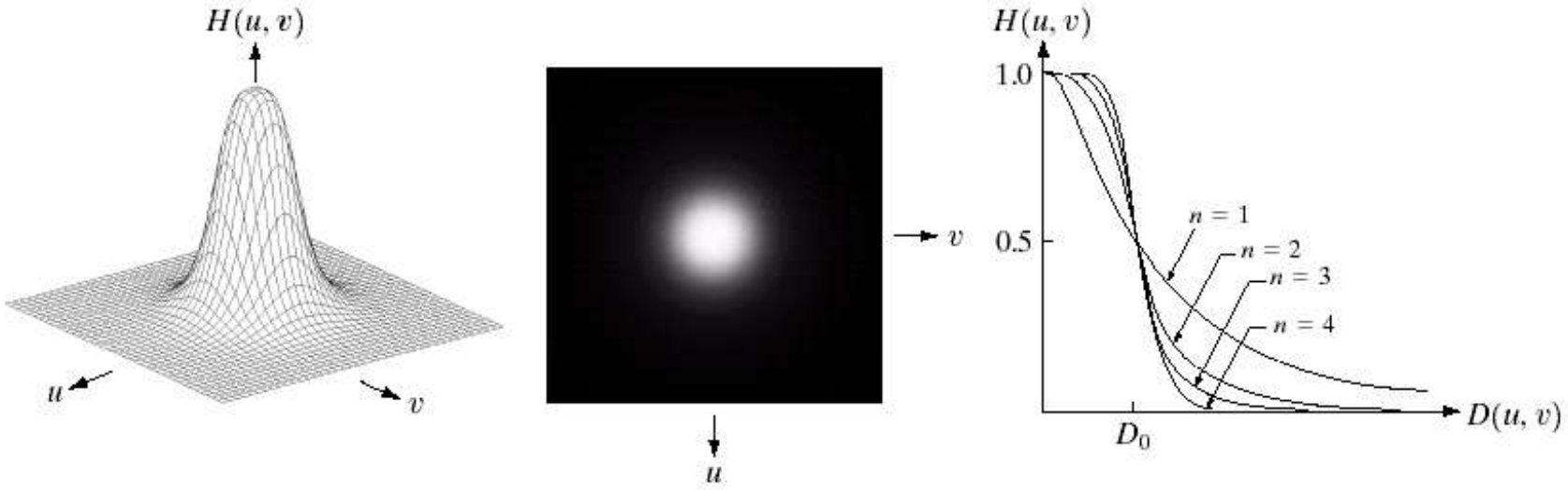


Low pass image

Butterworth Lowpass Filters

Definition:

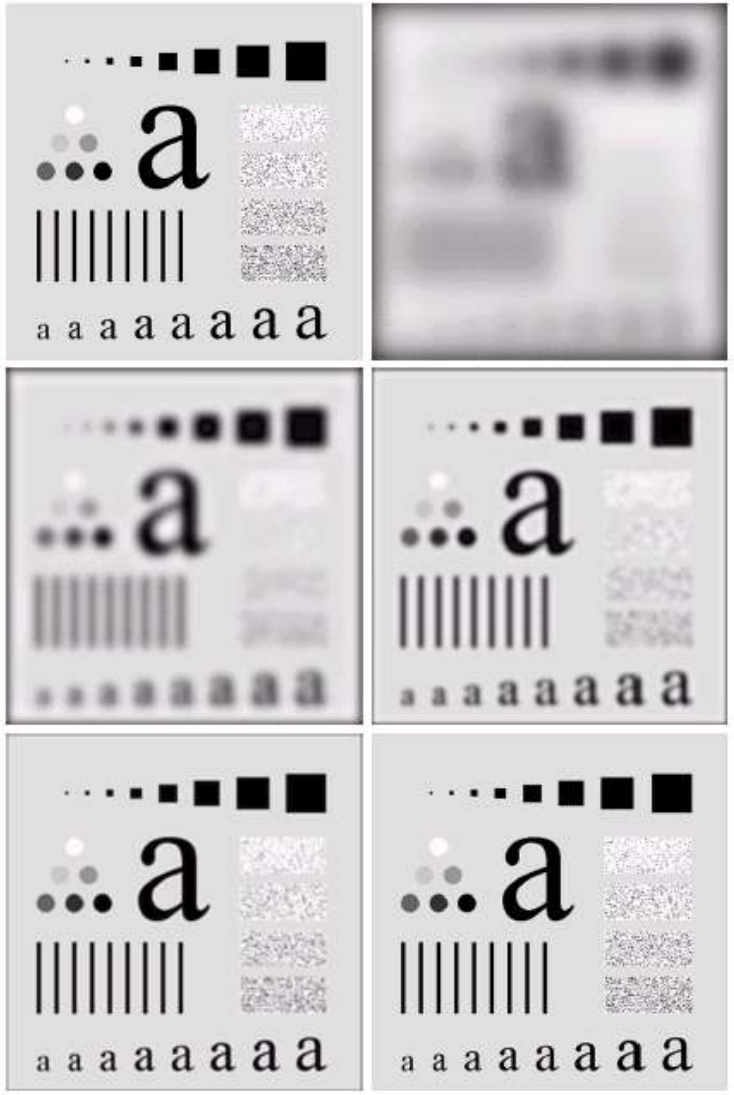
$$H(u,v) = \frac{1}{1+\left[D(u,v)/D_0\right]^{2n}}$$



a b c

FIGURE 4.14 (a) Perspective plot of a Butterworth lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections of orders 1 through 4.

Example

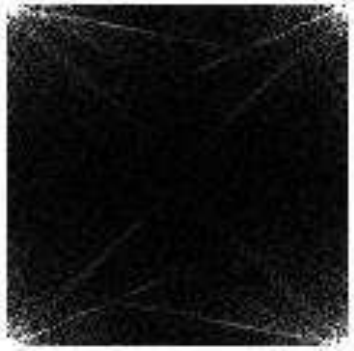


a b
c d
e f

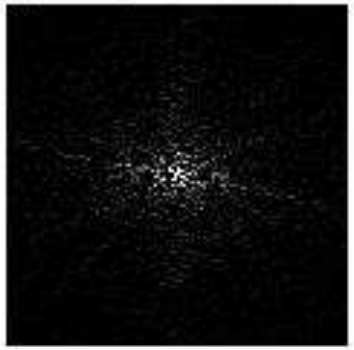
FIGURE 4.15 (a) Original image. (b)–(f) Results of filtering with BLPFs of order 2, with cutoff frequencies at radii of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). Compare with Fig. 4.12.

original image

F.T. of i/p without shift

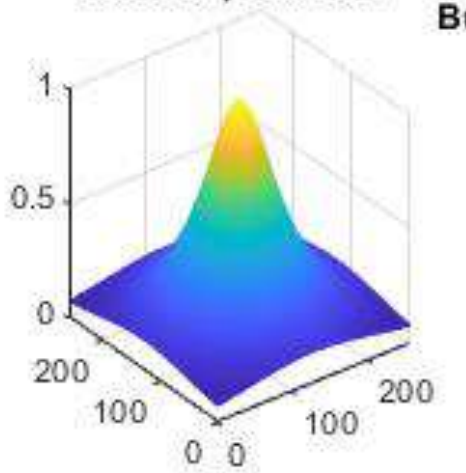


Frequency domain image



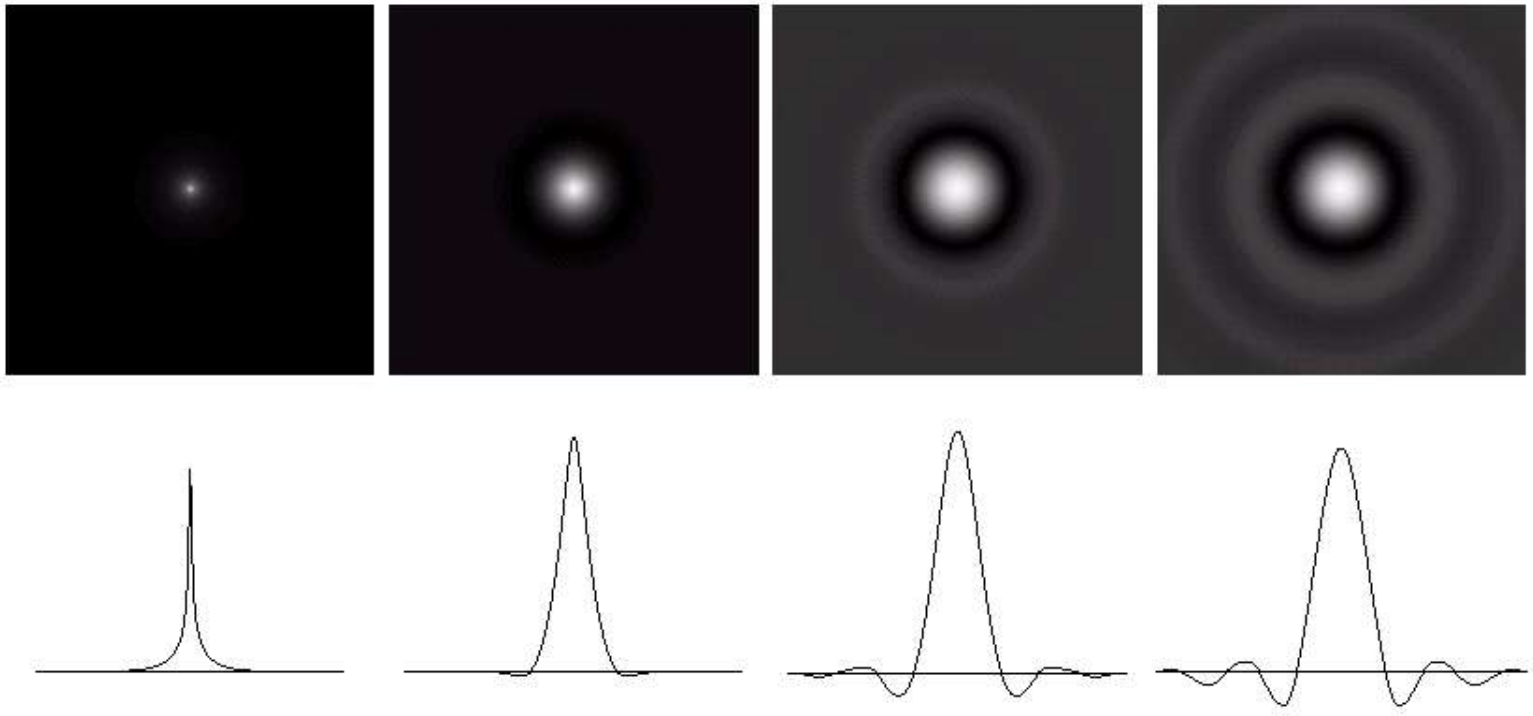
Butterworth LP Filter

surface plot BLPF



Butterworth LP Filtered image

Ringing Effect



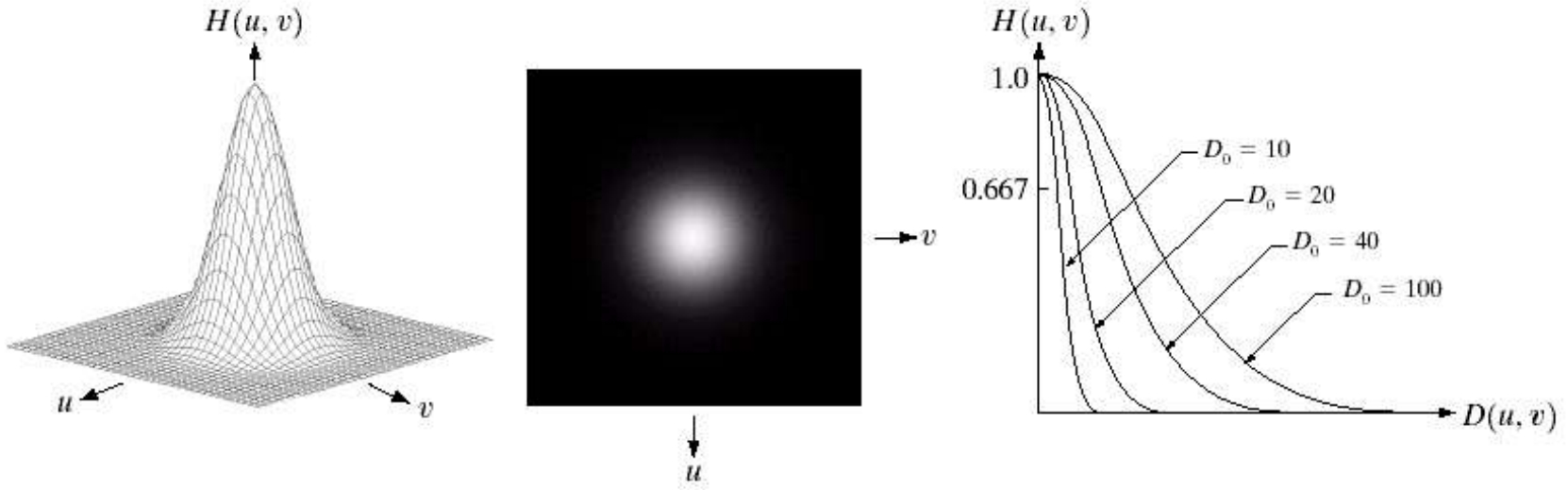
a b c d

FIGURE 4.16 (a)–(d) Spatial representation of BLPFs of order 1, 2, 5, and 20, and corresponding gray-level profiles through the center of the filters (all filters have a cutoff frequency of 5). Note that ringing increases as a function of filter order.

Gaussian Lowpass Filters

Definition:

$$H(u,v) = e^{-D^2(u,v)/2\sigma^2}$$



a b c

FIGURE 4.17 (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of D_0.

Example

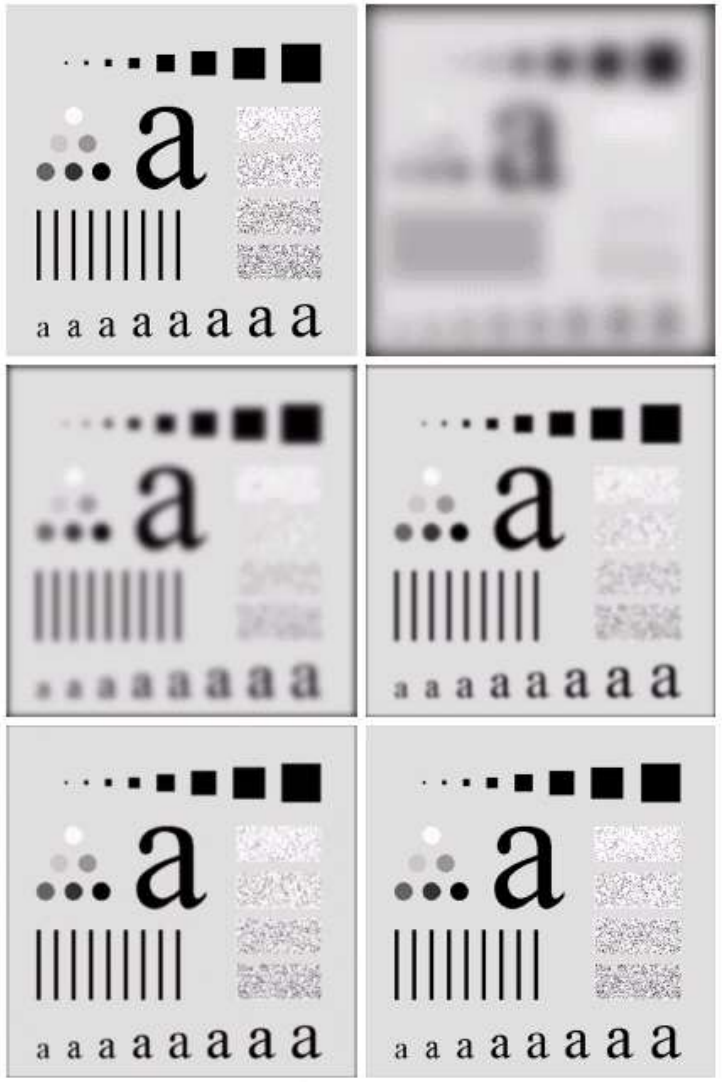
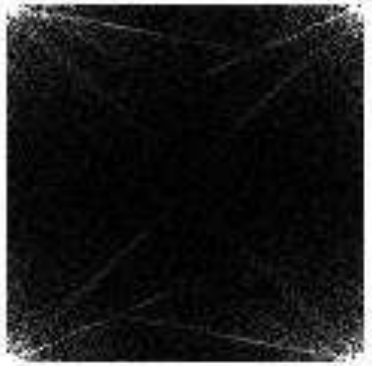


FIGURE 4.18 (a) Original image. (b)–(f) Results of filtering with Gaussian lowpass filters with cutoff frequencies set at radii values of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). Compare with Figs. 4.12 and 4.15.

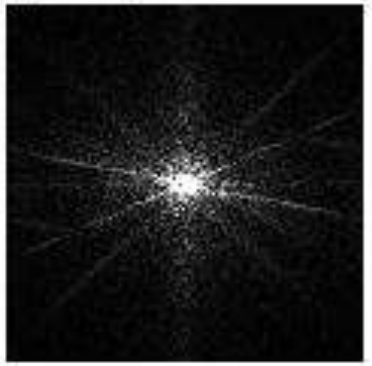
a b
c d
e f

Input image

F.T. of i/p without shift

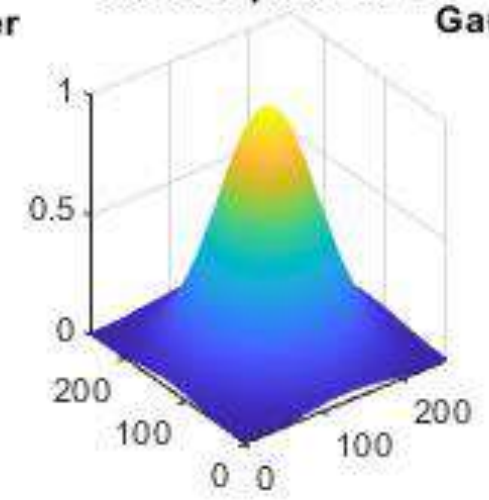


Frequency domain image



Gaussian Low pass Filter

surface plot GLPF

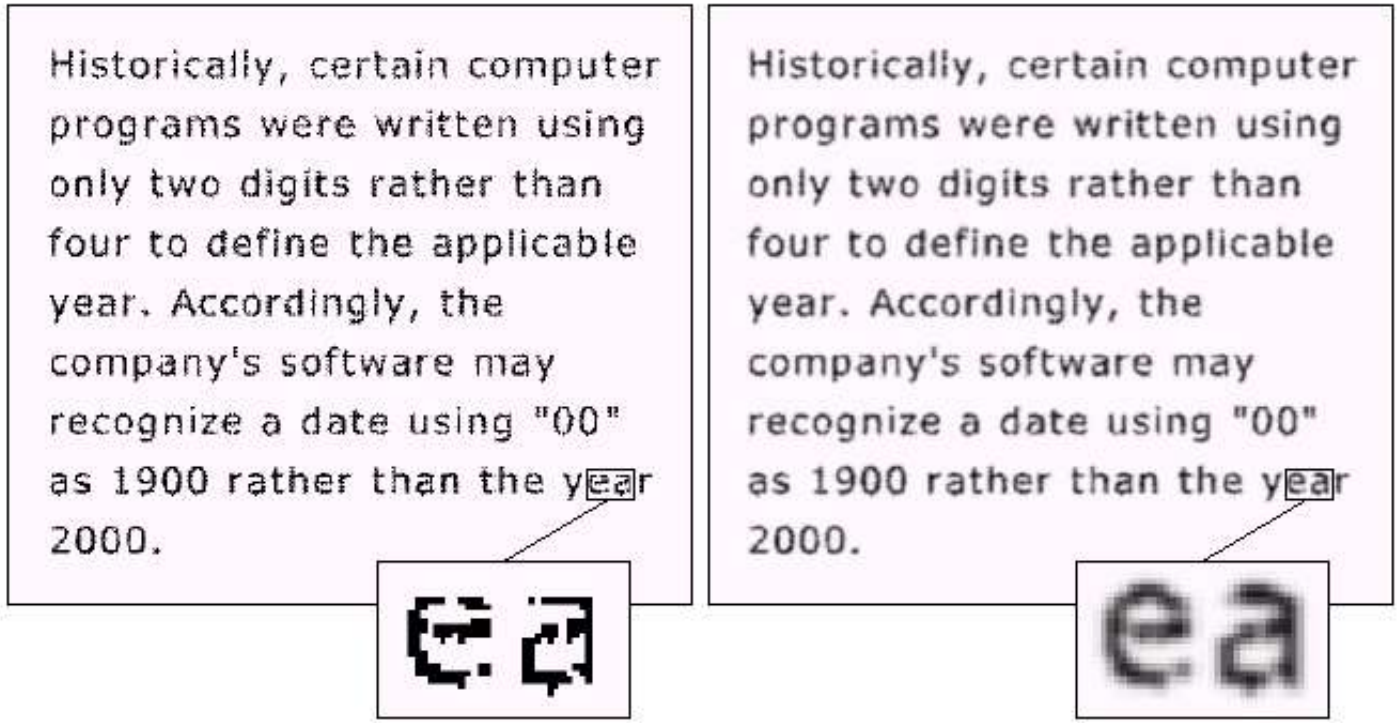


Gaussian Low pass Filtered image

More example

a b

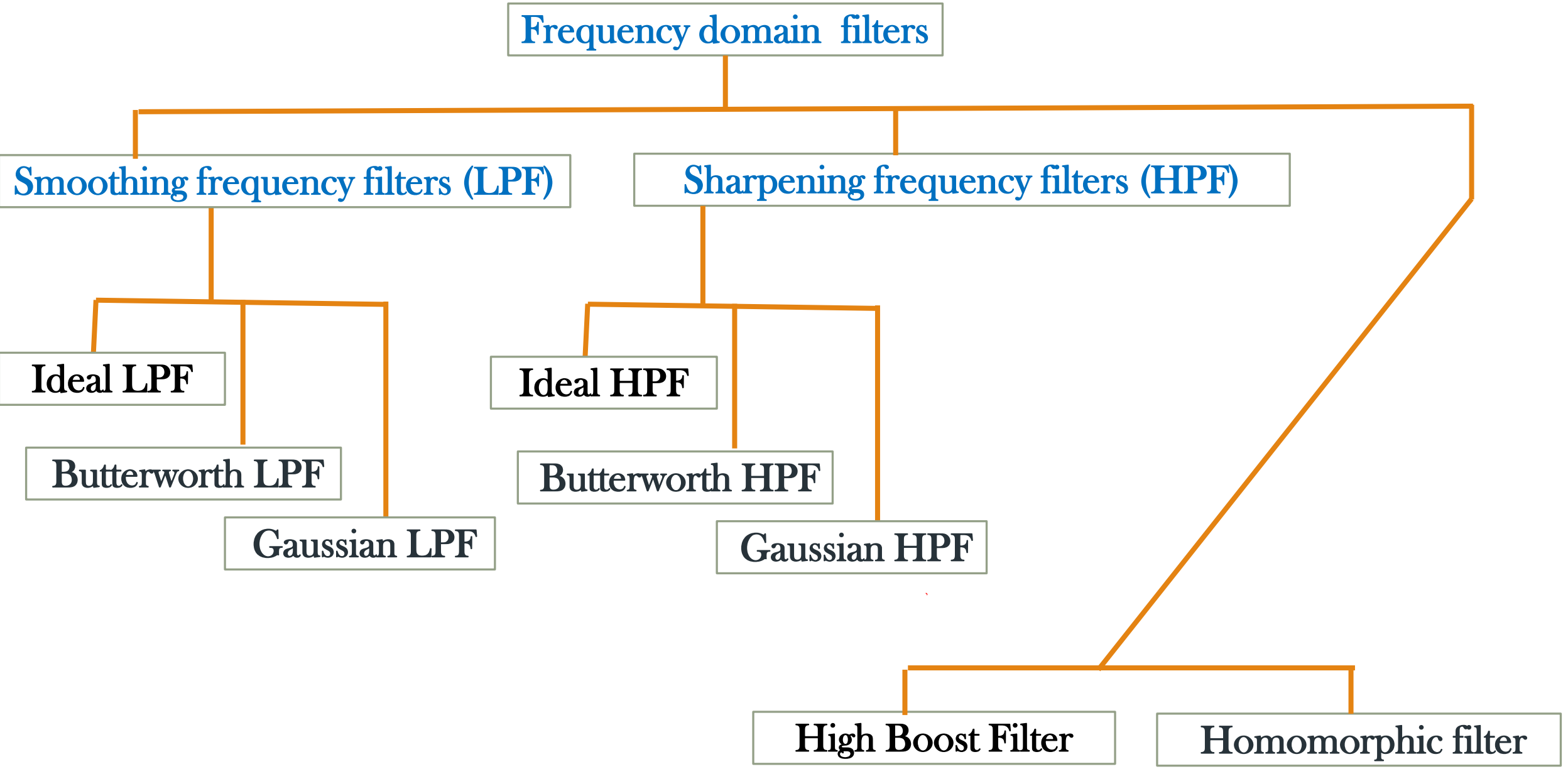
FIGURE 4.19
(a) Sample text of poor resolution (note broken characters in magnified view). (b) Result of filtering with a GLPF (broken character segments were joined).



Thank You

20

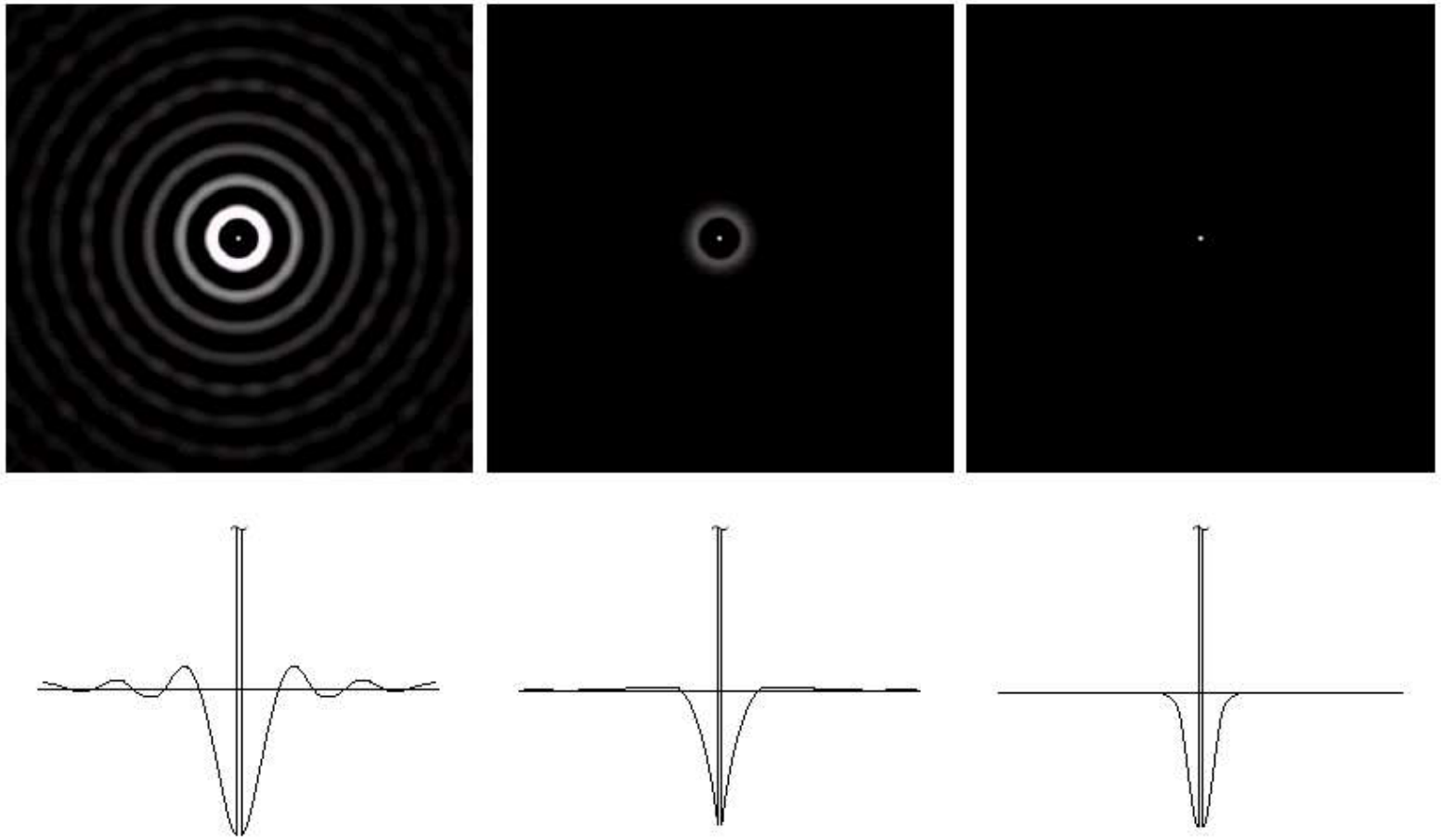
Image Sharpening (HPF) in frequency domain filtering and its Implementation in MATLAB


Frequency domain filters
Smoothing frequency filters (LPF)
Sharpening frequency filters (HPF)
Ideal LPF
Butterworth LPF
Gaussian LPF
Ideal HPF
Butterworth HPF
Gaussian HPF
High Boost Filter
Homomorphic filter

❖ Sharpening Filters

Ideal highpass filters	$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \le D_0 \\ 1 & \text{otherwise} \end{cases}$	This has a sharp discontinuity and hence, Ringing effect is present
Butterworth highpass filters	$H(u,v) = \dfrac{1}{1 + [D_0/D(u,v)]^{2n}}$	Does not have a sharp discontinuity and hence ringing is only observed for filters of higher orders.
Gaussian highpass filters	$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$	The profile of GHPF is not as tight as that of BLPF and hence no ringing effect is present.

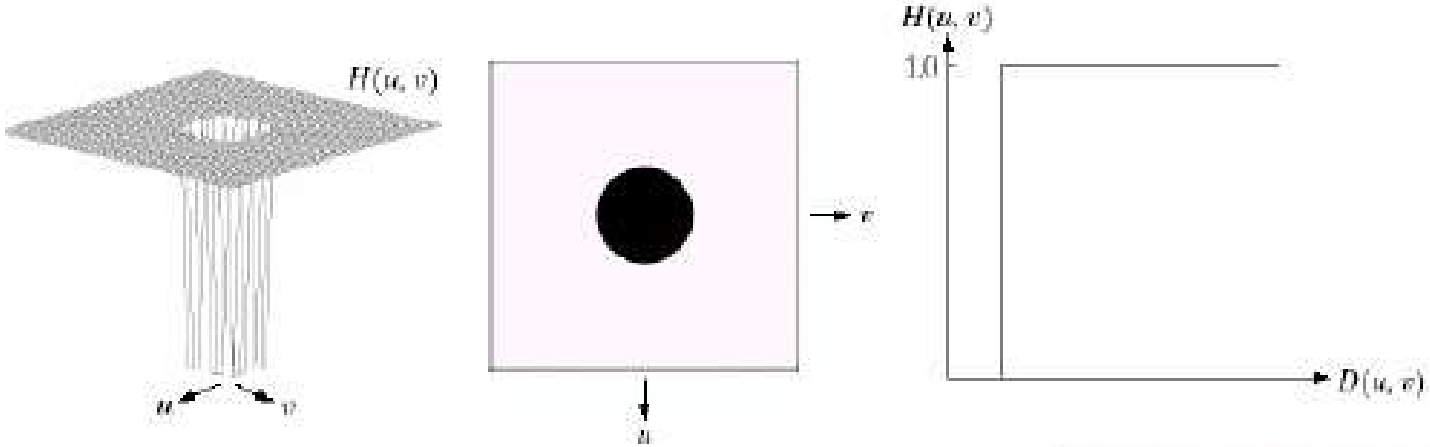
Spatial Representation of Freq. Domain HPF

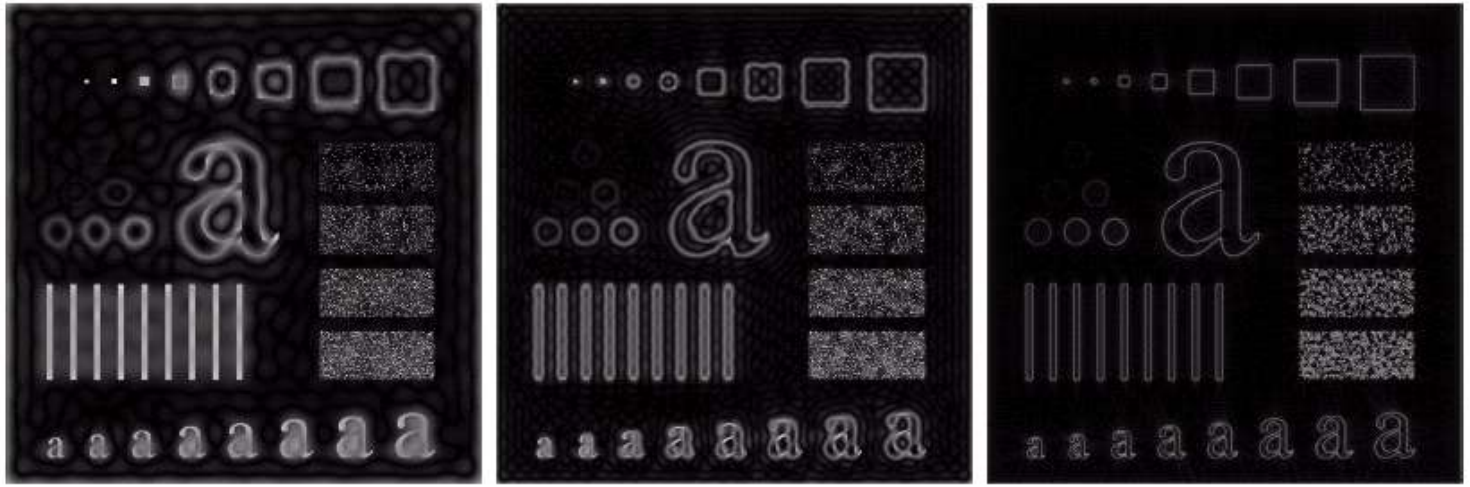


a b c

FIGURE 4.23 Spatial representations of typical (a) ideal, (b) Butterworth, and (c) Gaussian frequency domain highpass filters, and corresponding gray-level profiles.

Ideal High Pass Example





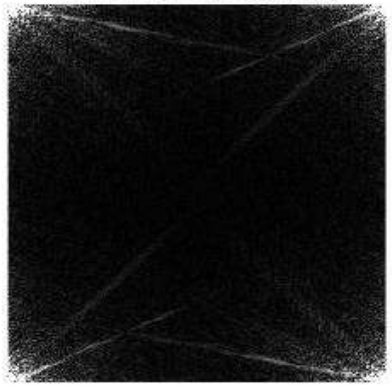
a b c

FIGURE 4.24 Results of ideal highpass filtering the image in Fig. 4.11(a) with $D_0 = 15$, 30, and 80, respectively. Problems with ringing are quite evident in (a) and (b).

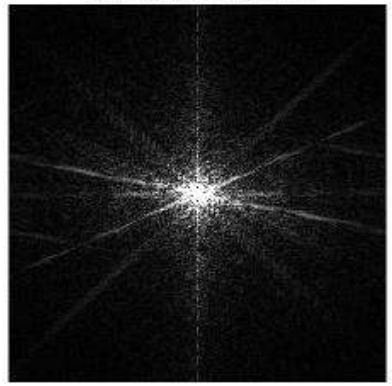
Ideal High Pass (Sharpening) Filters IHPF:

Input Image

F.T. of i/p without shift

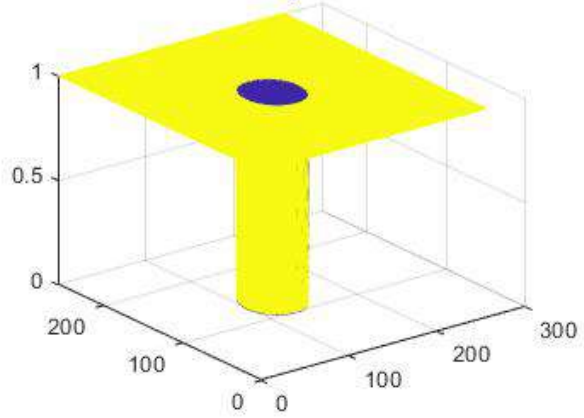


F.T. of i/p after shift



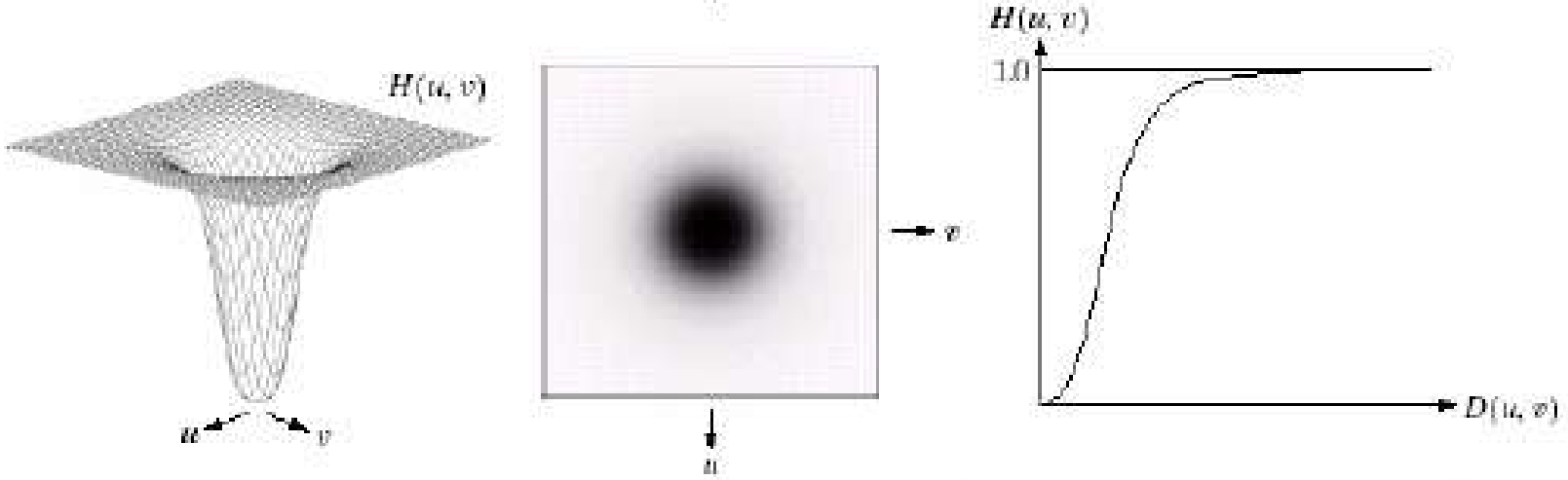
Ideal High pass filter

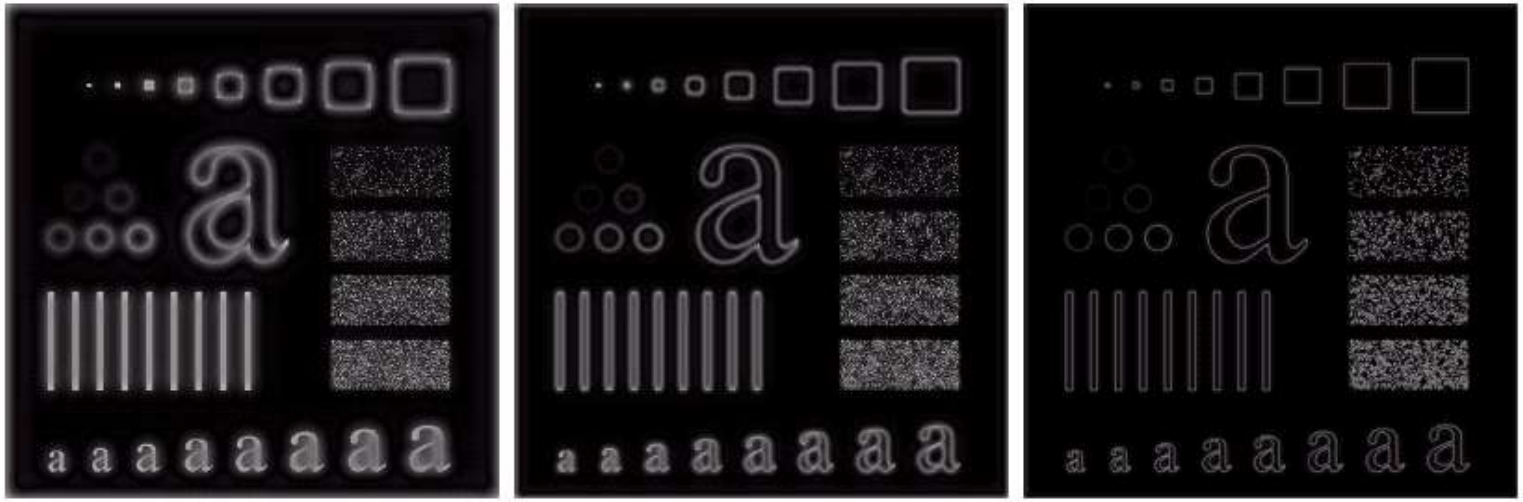
surface plot HPF



High pass image

Butterworth High Pass Example





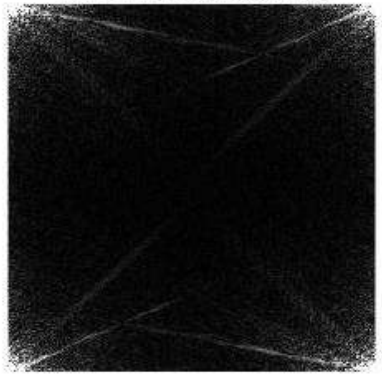
a b c

FIGURE 4.25 Results of highpass filtering the image in Fig. 4.11(a) using a BHPF of order 2 with $D_0 = 15$, 30, and 80, respectively. These results are much smoother than those obtained with an ILPF.

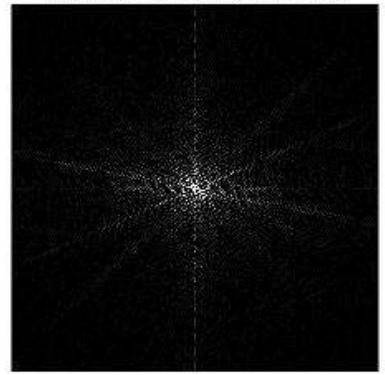
Butterworth High Pass (Sharpening) Filters BHPF:

Input image

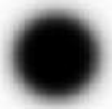
F.T. of i/p without shift



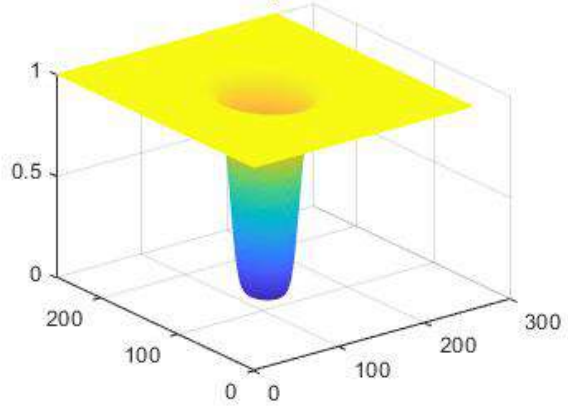
Frequency domain image



Butterworth HP Filter

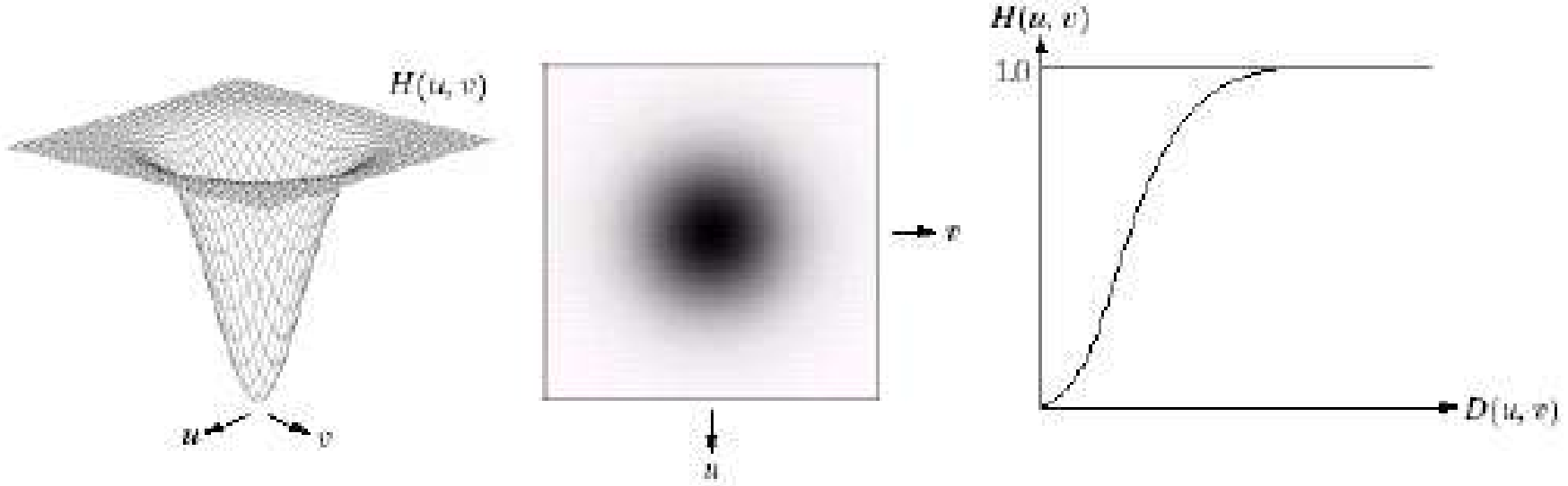


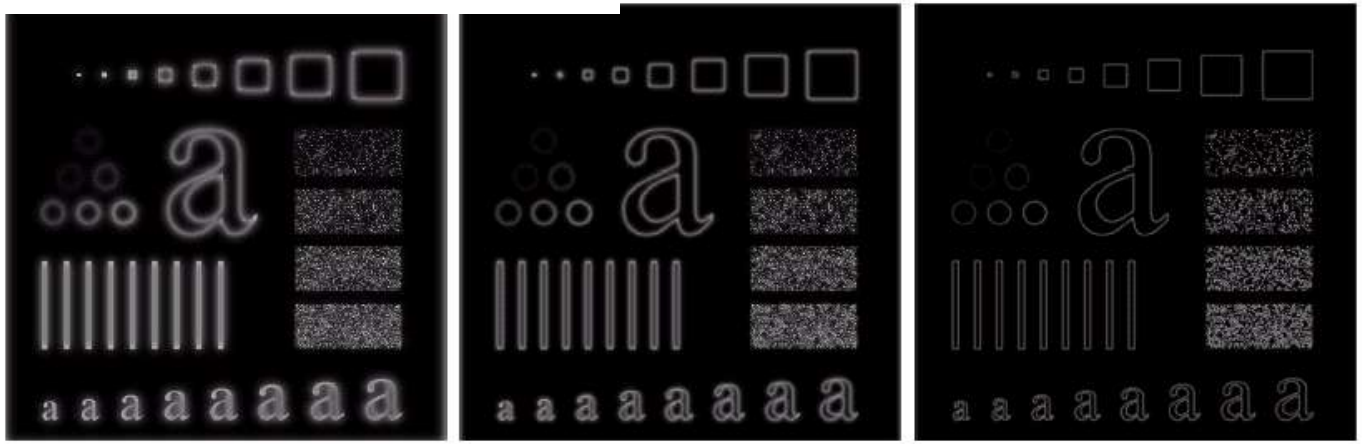
surface plot BHPF



Butterworth HP Filtered image

Gaussian High Pass Example





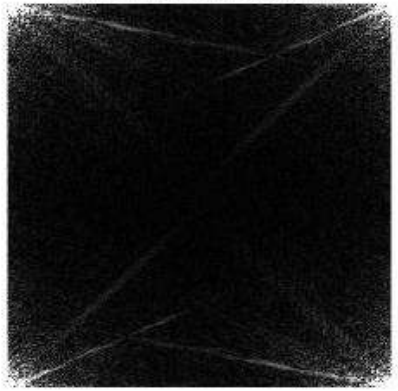
a b c

FIGURE 4.26 Results of highpass filtering the image of Fig. 4.11(a) using a GHPF of order 2 with $D_0 = 15$, 30, and 80, respectively. Compare with Figs. 4.24 and 4.25.

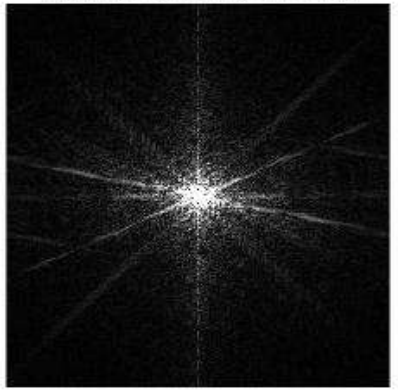
Gaussian High Pass (Sharpening) Filters GHPF:

Input image

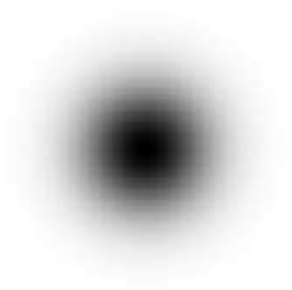
F.T. of i/p without shift



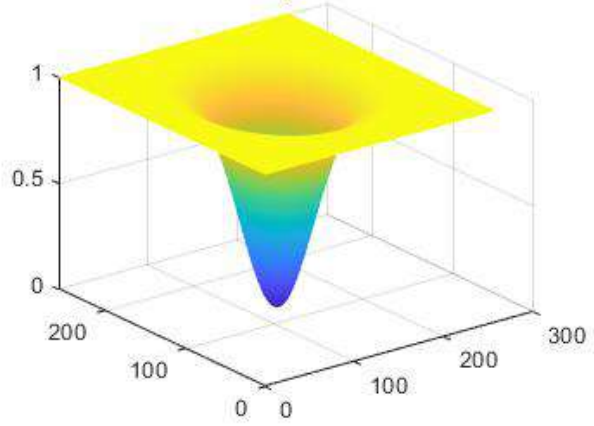
Frequency domain image



Gaussian High pass Filter



surface plot GHPF

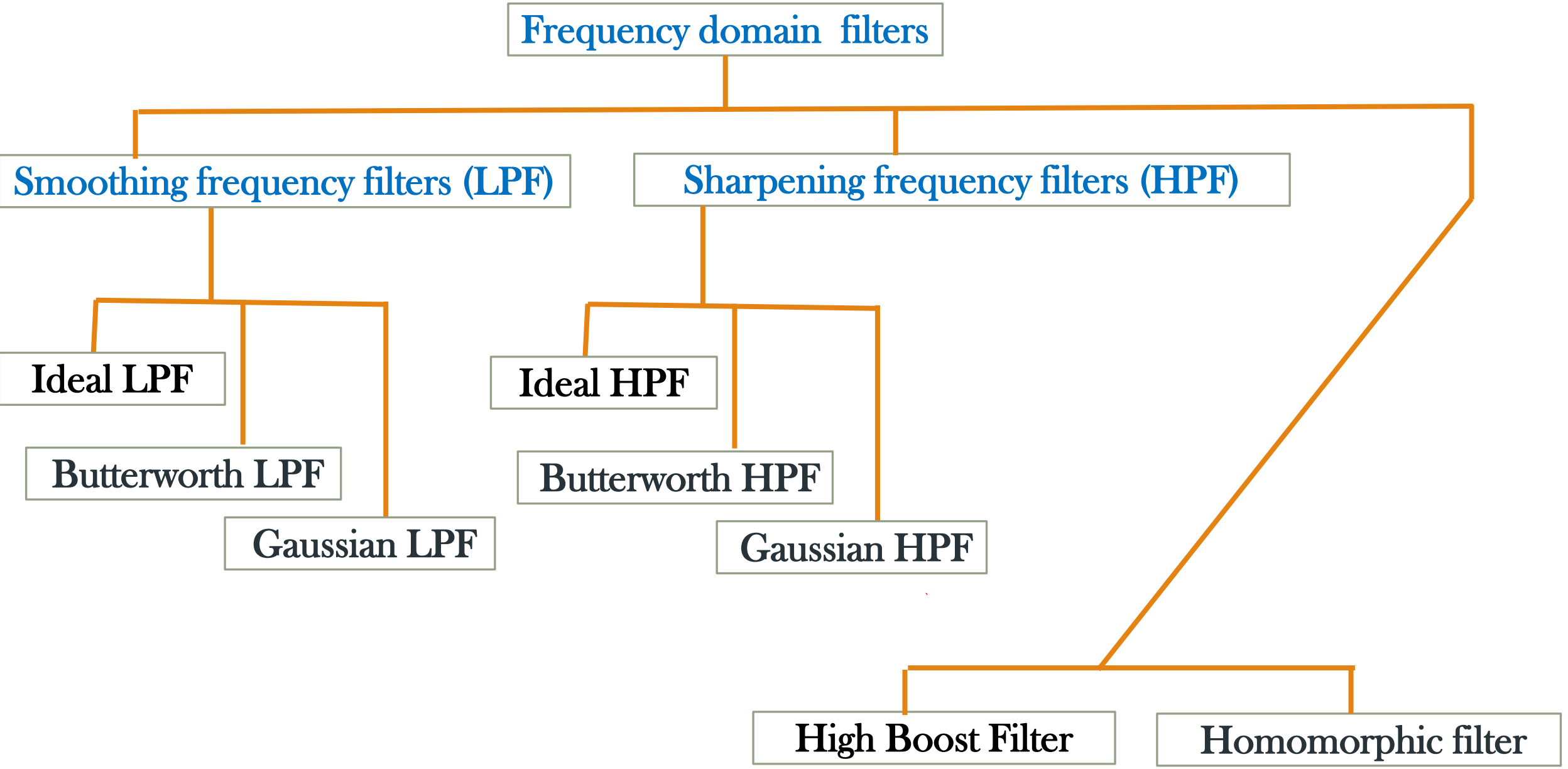


Gaussian High pass Filtered image

Thank You

21

Laplacian, Unsharp masking/High Boost filtering in the frequency domain filtering and its Implementation in MATLAB


Frequency domain filters
Smoothing frequency filters (LPF)
Sharpening frequency filters (HPF)
Ideal LPF
Butterworth LPF
Gaussian LPF
Ideal HPF
Butterworth HPF
Gaussian HPF
High Boost Filter
Homomorphic filter

❖ The Laplacian in the Spatial domain:

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

0	1	0	1	1	1
1	-4	1	1	-8	1
0	1	0	1	1	1

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

❖ The Laplacian in the Frequency domain:

$$\mathcal{F}\left[\frac{d^n f(x)}{dx^n}\right] = (ju)^n F(u)$$

Then,

$$\mathcal{F}\left[\frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}\right] = (ju)^2 F(u,v) + (jv)^2 F(u,v)$$
$$= -(u^2 + v^2) F(u,v)$$

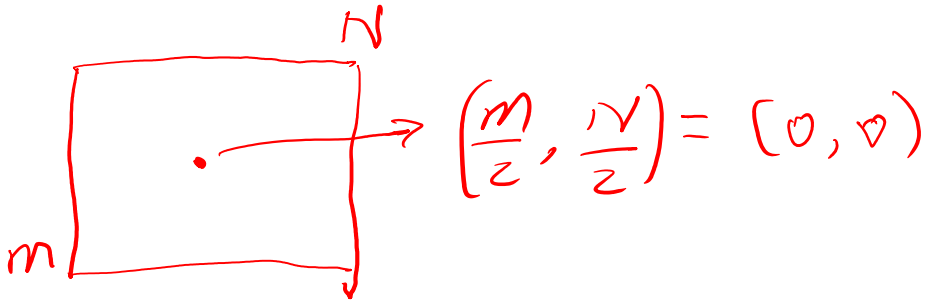
Finally,

$$\mathcal{F}[\nabla^2 f(x,y)] = -(u^2 + v^2) F(u,v)$$

G(u,v) = H(u,v) × F(u,v)

The Laplacian filter in the frequency domain is:

$$H(u,v) = -(u^2 + v^2)$$



If $F(u, v)$ has been centered by $f(x, y)(-1)^{x+y}$

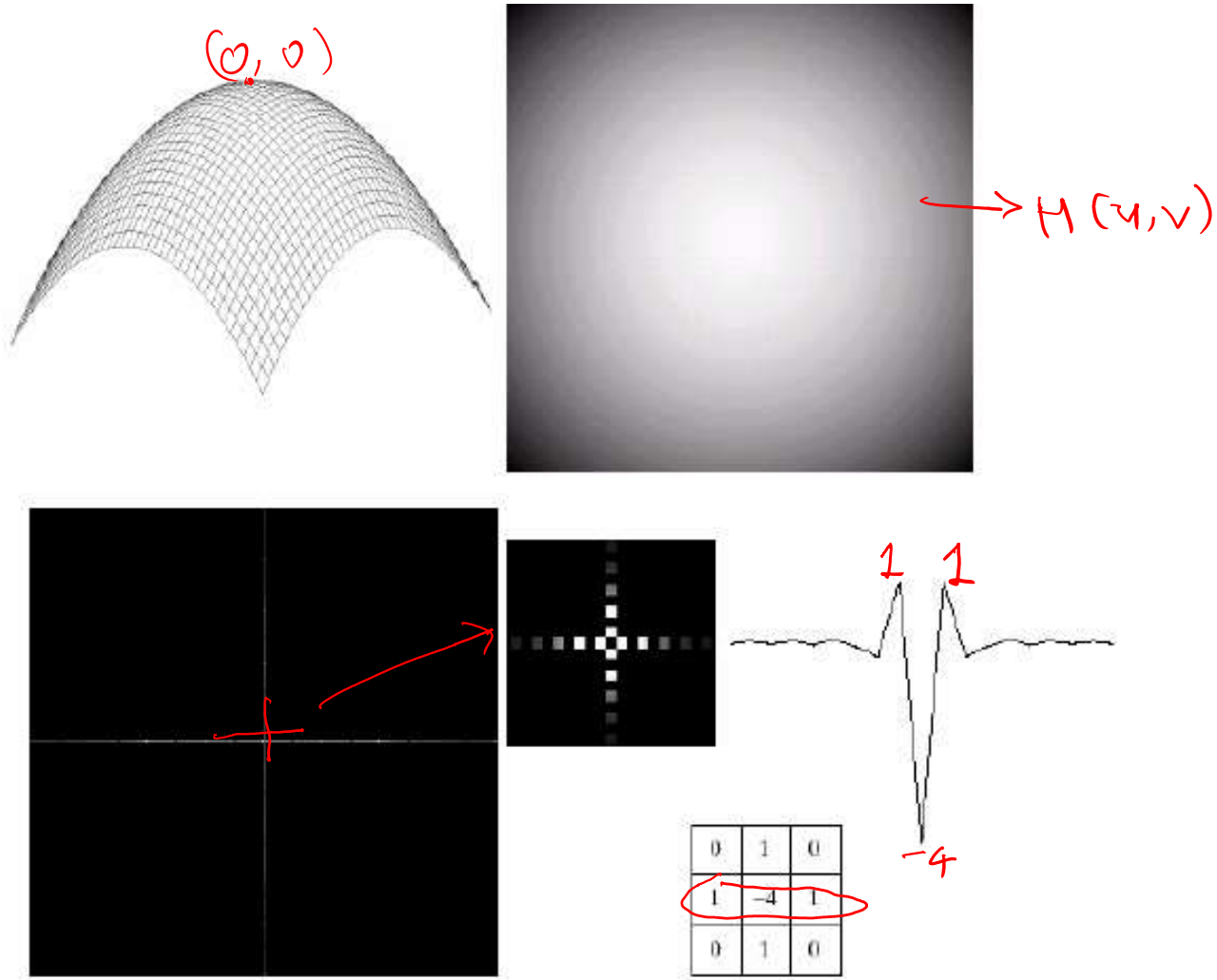
then $H(u, v) = -[(u-M/2)^2+(v-N/2)^2]$

- Laplacian filtering

$$\nabla^2 f(x,y) = \mathcal{F}^{-1}\{-[(u - M/2)^2 + (v - N/2)^2]F(u,v)\}$$

- Enhanced image $g(x, y) = f(x, y) - \nabla^2 f(x, y)$

$$\nabla^2 f(x,y) \Leftrightarrow -[(u - M/2)^2 + (v - N/2)^2]F(u,v).$$

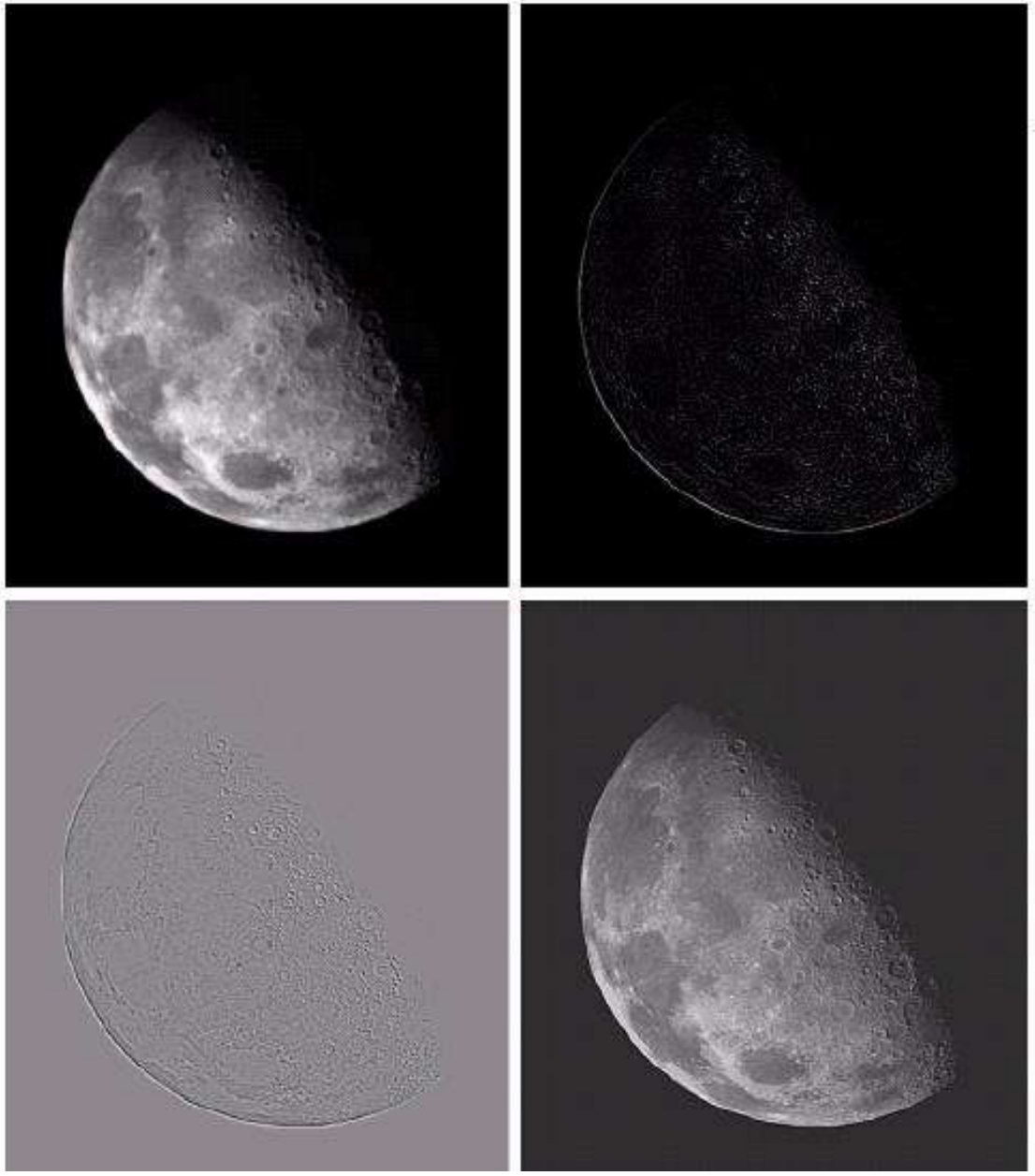


a b
c d e
f

FIGURE 4.27 (a) 3-D plot of Laplacian in the frequency domain. (b) Image representation of (a). (c) Laplacian in the spatial domain obtained from the inverse DFT of (b). (d) Zoomed section of the origin of (c). (e) Gray-level profile through the center of (d). (f) Laplacian mask used in Section 3.7.

a b
c d

FIGURE 4.28
(a) Image of the North Pole of the moon.
(b) Laplacian filtered image.
(c) Laplacian image scaled.
(d) Image enhanced by using Eq. (4.4-12).
(Original image courtesy of NASA.)



❖ High boost filtering in the Frequency domain:

HP = 1 − LP

Sometimes it is advantageous to increase the contribution made by the original image to the overall filtered result. This approach, called high-boost filtering, is a generalization of unsharp masking. Unsharp masking consists simply of generating a sharp image by subtracting from an image a blurred version of itself. Using frequency domain terminology, this means obtaining a highpass-filtered image by subtracting from the image a lowpass-filtered version of itself.

$$f_{hp}(x, y) = f(x, y) - f_{lp}(x, y)$$

High-boost filtering generalizes this by multiplying f (x, y) by a constant A > 1:

$$f_{hb} = Af(x, y) - f_{lp}(x, y)$$

$+f(x,y) - f(x,y) = Af(x,y) - f(x,y) + f(x,y) - f_{lp}(x,y)$

Thus, high-boost filtering gives us the flexibility to increase the contribution made by the image to the overall enhanced result. This equation may be written as:

$$f_{hb}(x, y) = (A - 1)f(x, y) + f(x, y) - f_{lp}(x, y)$$

Then, using above Eq. we obtain

$$f_{hb}(x, y) = (A - 1)f(x, y) + f_{hp}(x, y)$$

This result is based on a highpass rather than a lowpass image. When A = 1, high-boost filtering reduces to regular highpass filtering. As A increases past 1, the contribution made by the image itself becomes more dominant.

We have $F_{hp}(u,v) = F(u,v) - F_{lp}(u,v)$. But $F_{lp}(u,v) = H_{lp}(u,v)F(u,v)$, where H_{lp} is the transfer function of a lowpass filter. Therefore, unsharp masking can be implemented directly in the frequency domain by using the composite filter

$$H_{hp}(u, v) = 1 - H_{lp}(u, v)$$

$f_{hb}(x,y) = (A-1)f(x,y) + f_{hp}(x,y)$

$F_{HB}(u,v) = (A-1)F(u,v) + F_{HP}(u,v)$

$F_{HB}(u,v) = (A-1)F(u,v) + H_{HP}(u,v) * F(u,v)$

Similarly, high-boost filtering can be implemented with the composite filter

$$H_{hb}(u, v) = (A - 1) + H_{hp}(u, v)$$

$F(u,v)H_{HB}(u,v) = F(u,v)[(A-1) + H_{HP}(u,v)]$

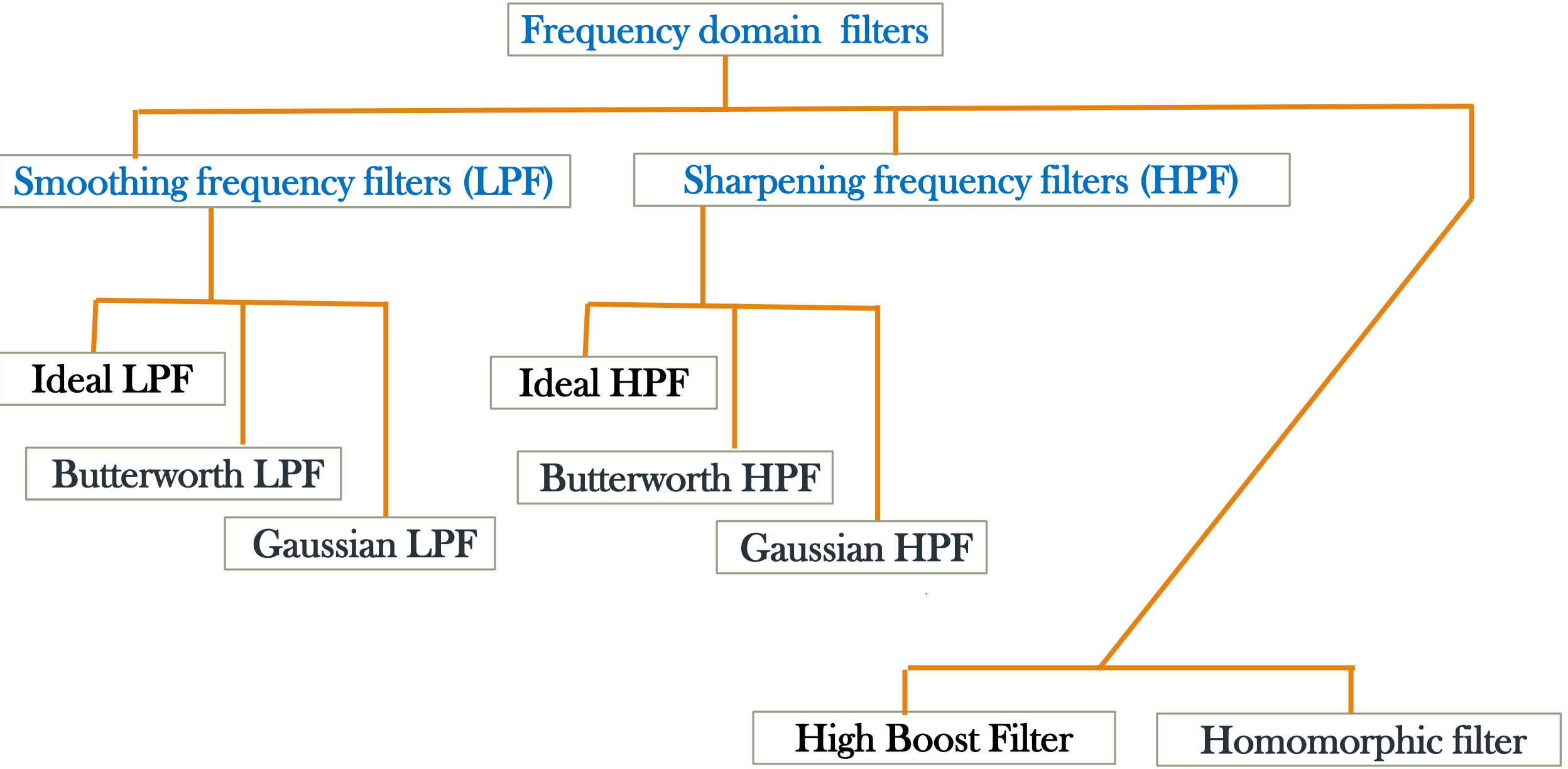
$\therefore H_{HB}(u,v) = (A-1) + H_{HP}(u,v) \; ; A > 1$

with A > 1. The process consists of multiplying this filter by the (centered) transform of the input image and then taking the inverse transform of the product. Multiplication of the real part of this result by $(-1)^{x+y}$ gives us the high-boost filtered image $f_{hb}(x, y)$ in the spatial domain.

Thank You

22

Homomorphic Filtering and its Implementation in MATLAB


Frequency domain filters
Smoothing frequency filters (LPF)
Sharpening frequency filters (HPF)
Ideal LPF
Butterworth LPF
Gaussian LPF
Ideal HPF
Butterworth HPF
Gaussian HPF
High Boost Filter
Homomorphic filter

❖ The Homomorphic Filtering:

- Homomorphic filters uses a different approach compared to other frequency domain filters. It normalizes the brightness across an image and increases contrast.
- Homomorphic filter can be used to enhance an image due to non-uniform illumination. Illumination variations can be thought of as a multiplication noise, and can be reduced by filtering in the log domain.
- To evenly distribute the illumination, the high-frequency components are increased and the low-frequency components are decreased, because the high frequency components are assumed to represent mostly the reflectance in the scene(the amount of light reflected off the object in the scene), whereases the low frequency components are assumed to represent mostly the illumination in the scene.
- Hence high pass filtering is used to suppress low frequencies and amplify high frequencies in the log-intensity domain.

- An image f(x, y) can be expressed as the product of illumination and reflectance components: $f(x, y) = i(x, y)r(x, y)$.
- Equation above cannot be used directly to operate separately on the frequency components of illumination and reflectance because the Fourier transform of the product of two functions is not separable; in other words,

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

Suppose, however, that we define

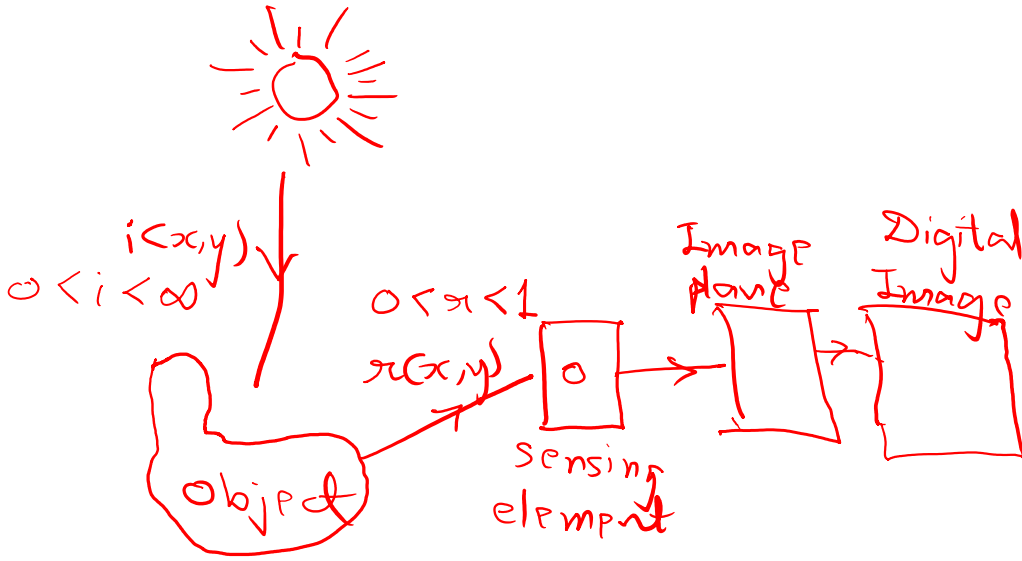
$$\begin{aligned} z(x, y) &= \ln f(x, y) \\ &= \ln i(x, y) + \ln r(x, y). \end{aligned}$$

Then

$$\begin{aligned} \Im\{z(x, y)\} &= \Im\{\ln f(x, y)\} \\ &= \Im\{\ln i(x, y)\} + \Im\{\ln r(x, y)\} \end{aligned}$$

or

$$Z(u, v) = F_i(u, v) + F_r(u, v)$$



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

$$\begin{aligned} S(u, v) &= H(u, v)Z(u, v) \\ &= H(u, v)F_i(u, v) + H(u, v)F_r(u, v) \end{aligned}$$

In the spatial domain,

$$\begin{aligned} s(x, y) &= \Im^{-1}\{S(u, v)\} \\ &= \Im^{-1}\{H(u, v)F_i(u, v)\} + \Im^{-1}\{H(u, v)F_r(u, v)\}. \end{aligned}$$

By letting

$$i'(x, y) = \Im^{-1}\{H(u, v)F_i(u, v)\}$$

and

$$r'(x, y) = \Im^{-1}\{H(u, v)F_r(u, v)\},$$

$$s(x, y) = i'(x, y) + r'(x, y).$$

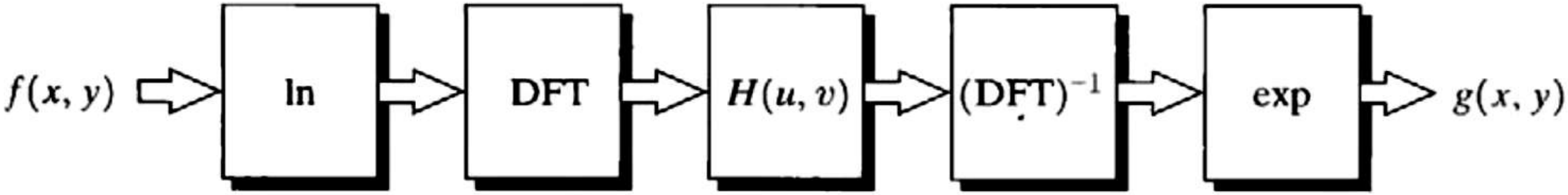
Finally, as z (x, y) was formed by taking the logarithm of the original image f (x, y), the inverse (exponential) operation yields the desired enhanced image, denoted by g(x, y); that is,

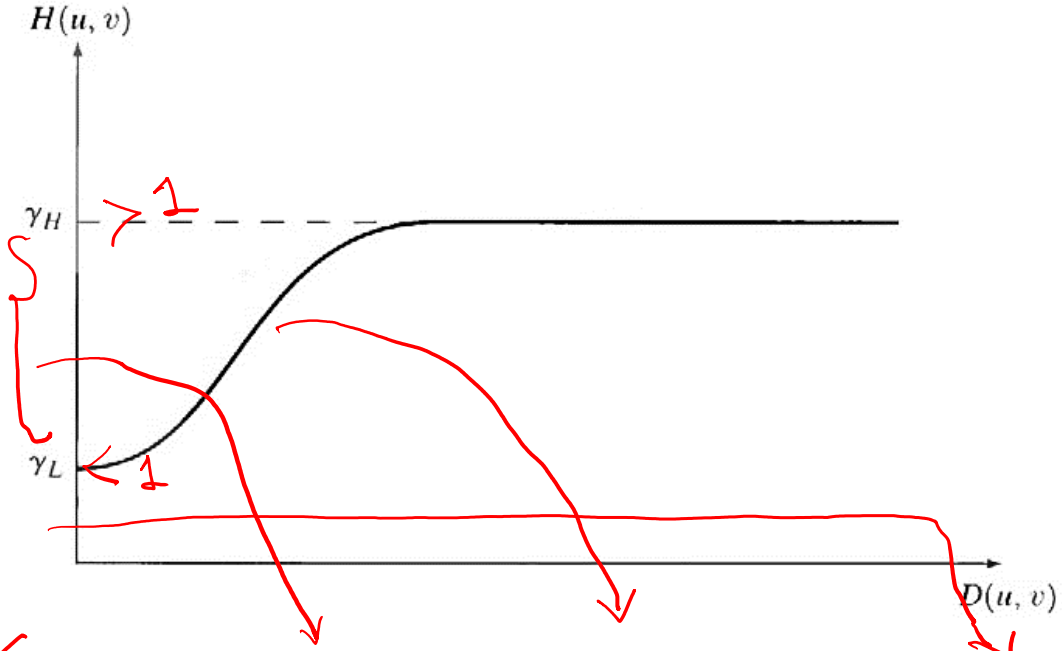
$$\begin{aligned} g(x, y) &= e^{s(x,y)} \\ &= e^{i'(x,y)} \cdot e^{r'(x,y)} \\ &= i_0(x, y)r_0(x, y) \end{aligned}$$

where

$$i_0(x, y) = e^{i'(x,y)} \qquad r_0(x, y) = e^{r'(x,y)}$$

Homomorphic filtering approach for image enhancement



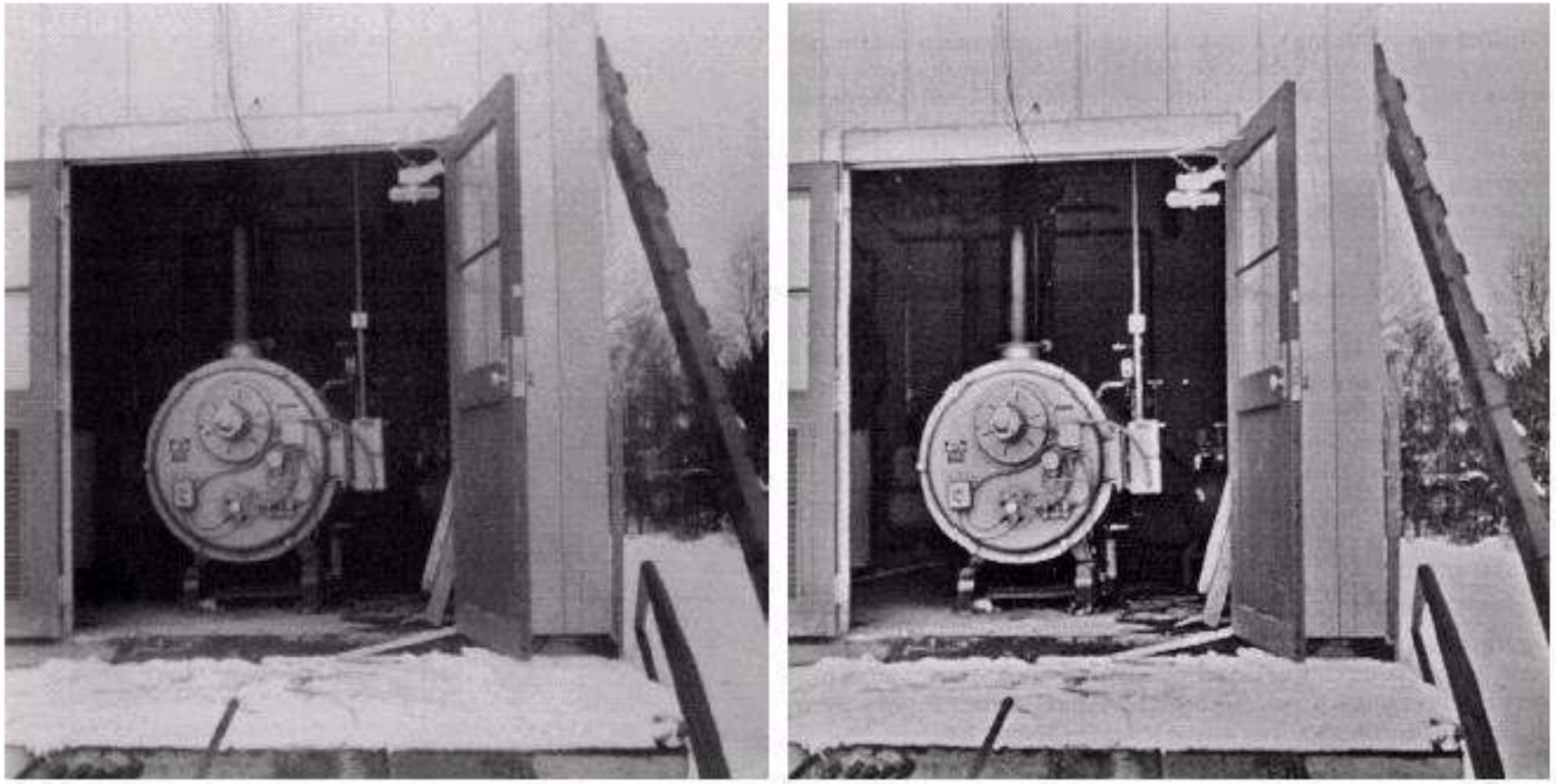


$$H(u,v) = (\gamma_H - \gamma_L)\left[1 - e^{-c[D^2(u,v)/D_0^2]}\right] + \gamma_L$$

If the parameters γ_L and γ_H are chosen so that $\gamma_L < 1$ and $\gamma_H > 1$, the filter function shown tends to decrease the contribution made by the low frequencies (illumination) and amplify the contribution made by high frequencies (reflectance). The net result is simultaneous dynamic range compression and contrast enhancement.

a b

FIGURE 4.33
(a) Original image. (b) Image processed by homomorphic filtering (note details inside shelter). (Stockham.)



Thank You