BME4120
Biomedical Image Processing

Lecture 4

Image Enhancement

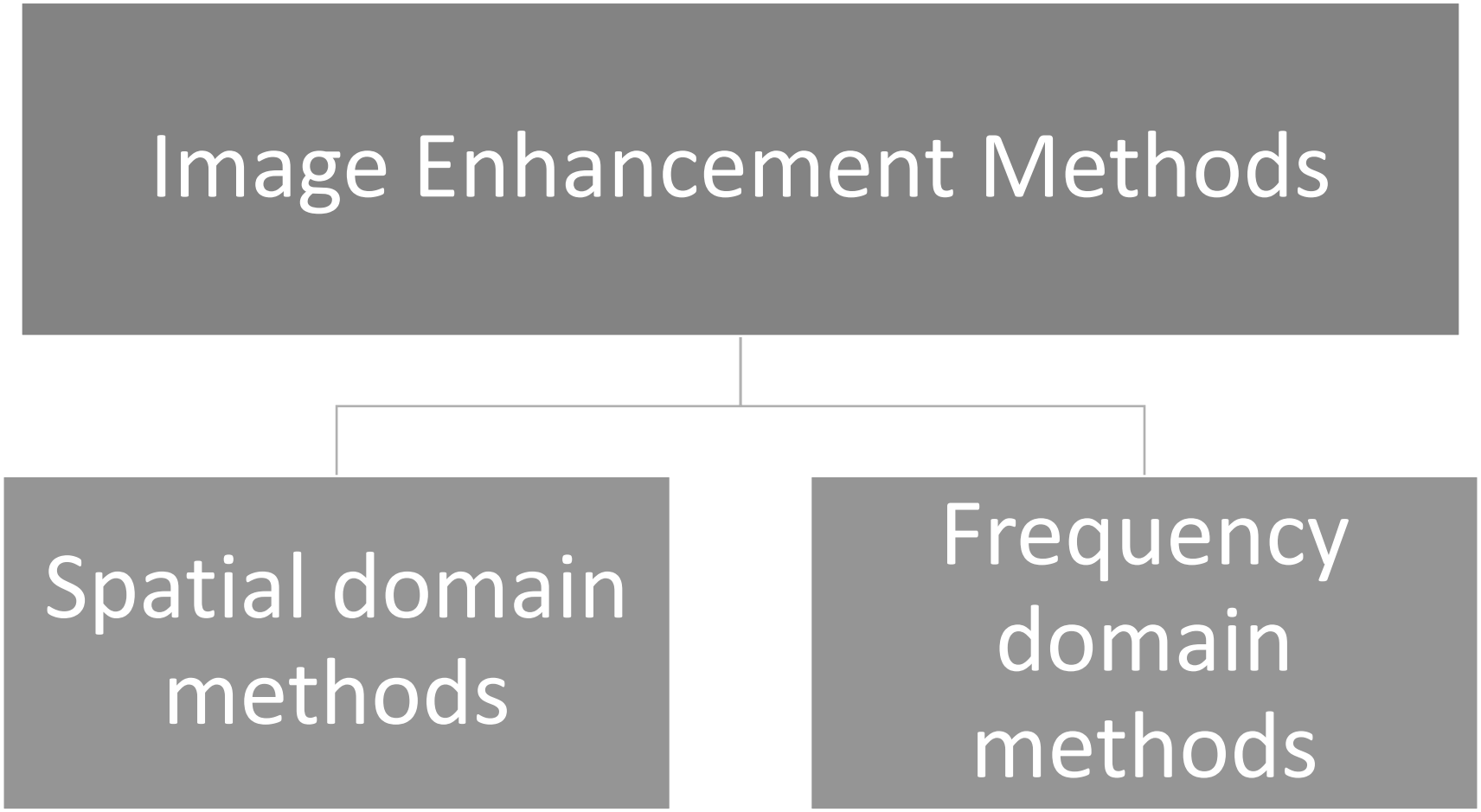
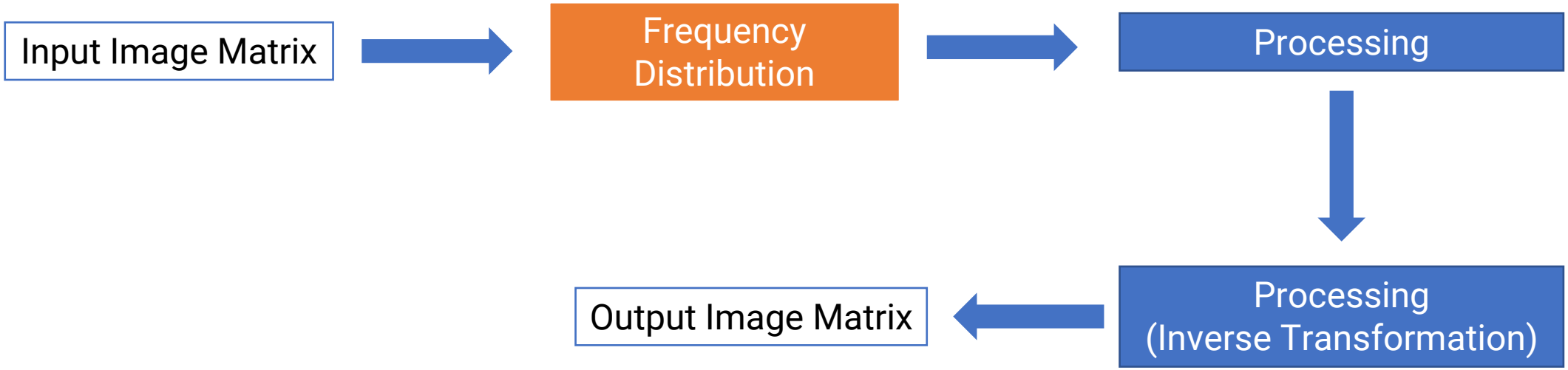


Image Enhancement

Spatial Domain

Frequency Domain



Frequency Distribution

Pixel intensity count for the entire image

Transformation to Frequency Domain

From spatial to frequency domain

- ❑ Fourier Series
- ❑ Fourier transformation
- ❑ Laplace transform
- ❑ Z transform
- ❑ …

Fourier Analysis

Fourier Analysis in a Nutshell

- Periodic signals can be represented as a sum of ***sines*** and ***cosines*** when multiplied by a certain weight (series),
- This series can be denoted by ***integral transformation***.

Fourier Series

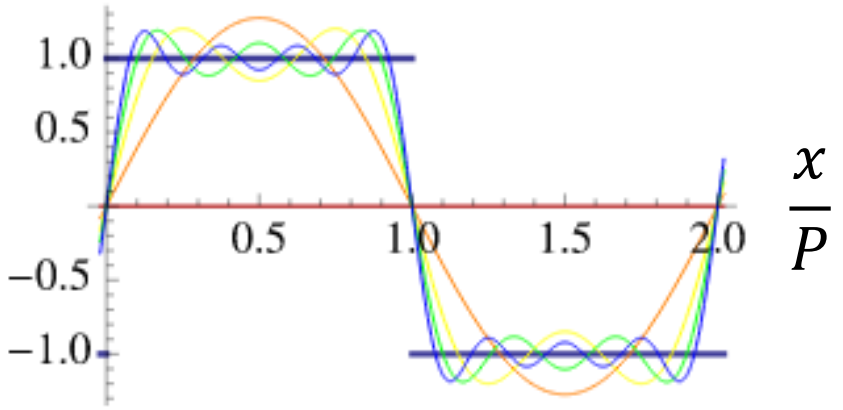
Periodic signals can be represented as a sum of sines and cosines when multiplied by a certain weight:

$$S(x) = \frac{a_0}{2} + \sum_{n=1}^{N} \left(a_n cos\left(\frac{2\pi nx}{P}\right) + b_n sin\left(\frac{2\pi nx}{P}\right) \right)$$

$$S(x) = \sum_{n=-N}^{N} c_n e^{i\frac{2\pi nx}{P}}$$

This also means periodic signals can be broken down into further signals with the following properties:

- ❑ The signals are sines and cosines
- ❑ The signals are harmonics of each other (i.e., their frequencies are integer multiples of other frequencies)



https://mathworld.wolfram.com/FourierSeries.html

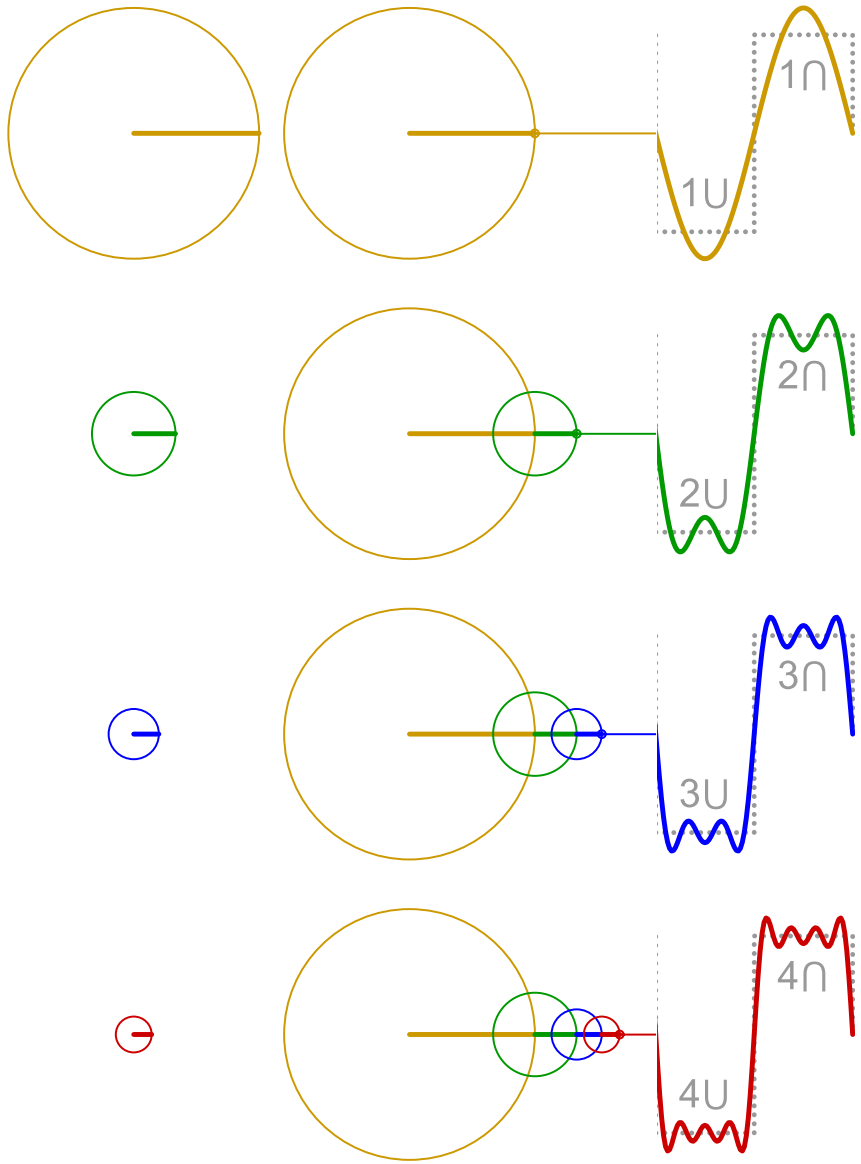
Fourier Series

$\frac{4}{\pi}\sin\theta$

$\frac{4}{3\pi}\sin 3\theta$

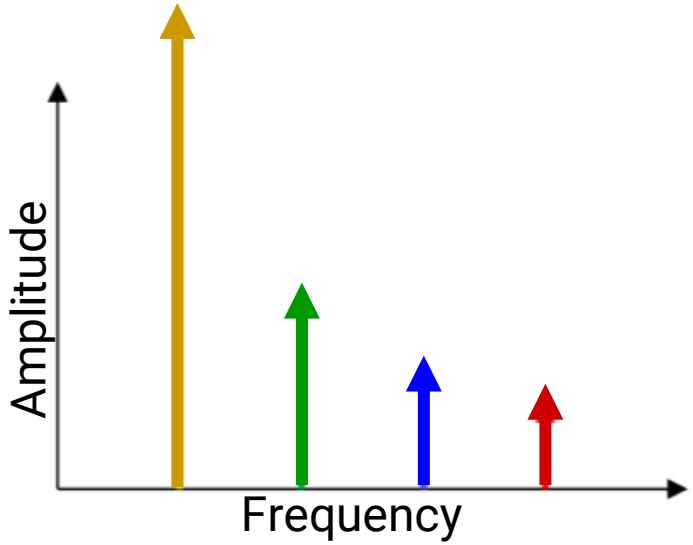
$\frac{4}{5\pi}\sin 5\theta$

$\frac{4}{7\pi}\sin 7\theta$



Four partial sums (Fourier series) of lengths 1, 2, 3, and 4 terms, showing how the approximation to a square wave improves as the number of terms increases.

Check the animation link below.



https://upload.wikimedia.org/wikipedia/commons/b/bd/Fourier_series_square_wave_circles_animation.svg

Fourier Series: Finding Coefficients

$$a_n = \int_0^P s(x)\cos\left(\frac{2\pi nx}{P}\right)dx$$

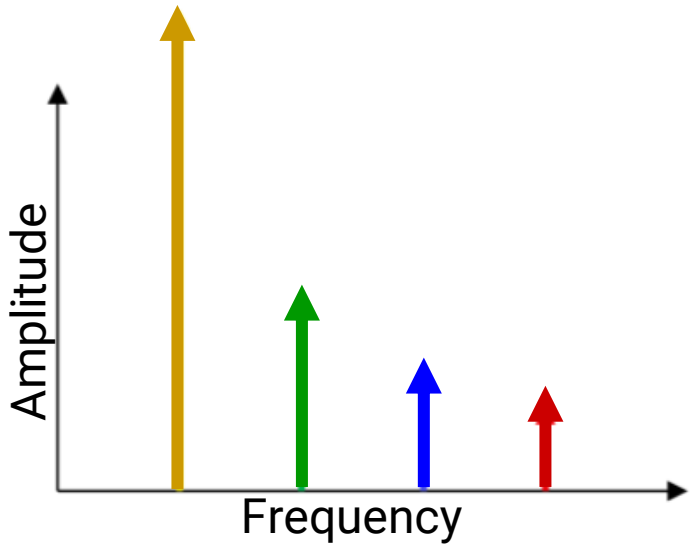
$$b_n = \int_0^P s(x)\sin\left(\frac{2\pi nx}{P}\right)dx$$

or alternatively,

$$c_n = \int_0^P s(x)e^{-i\frac{2\pi nx}{P}}dx$$

Four partial sums (Fourier series) of lengths 1, 2, 3, and 4 terms, showing how the approximation to a square wave improves as the number of terms increases.

Check the animation link below.



https://upload.wikimedia.org/wikipedia/commons/b/bd/Fourier_series_square_wave_circles_animation.svg

Fourier Transform

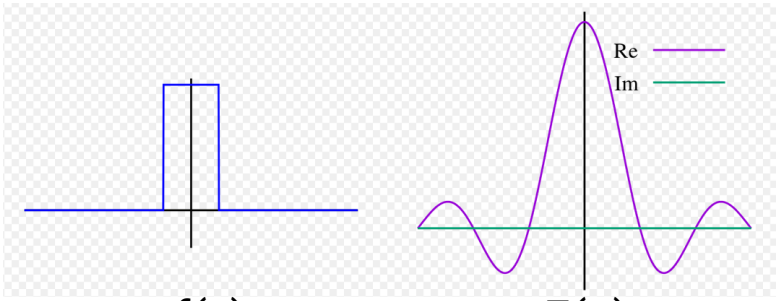
- Aperiodic signals can be decomposed into a continuum of sine waves.

→ Non-periodic signals whose area under the curve is finite can also be represented as integrals of the sines and cosines after being multiplied by a certain weight.

$$f(x) = \int_{-\infty}^{\infty} \mathrm{F}(u) \cdot e^{i2\pi ux} dx$$

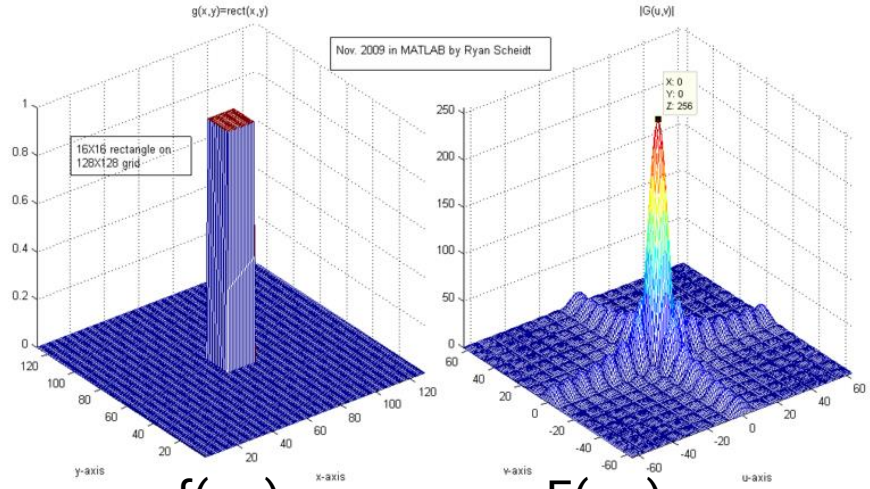
or multidimensional

$$f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mathrm{F}(u,v) \cdot e^{i2\pi(ux+vy)} dx\, dy$$



f(x) F(u)

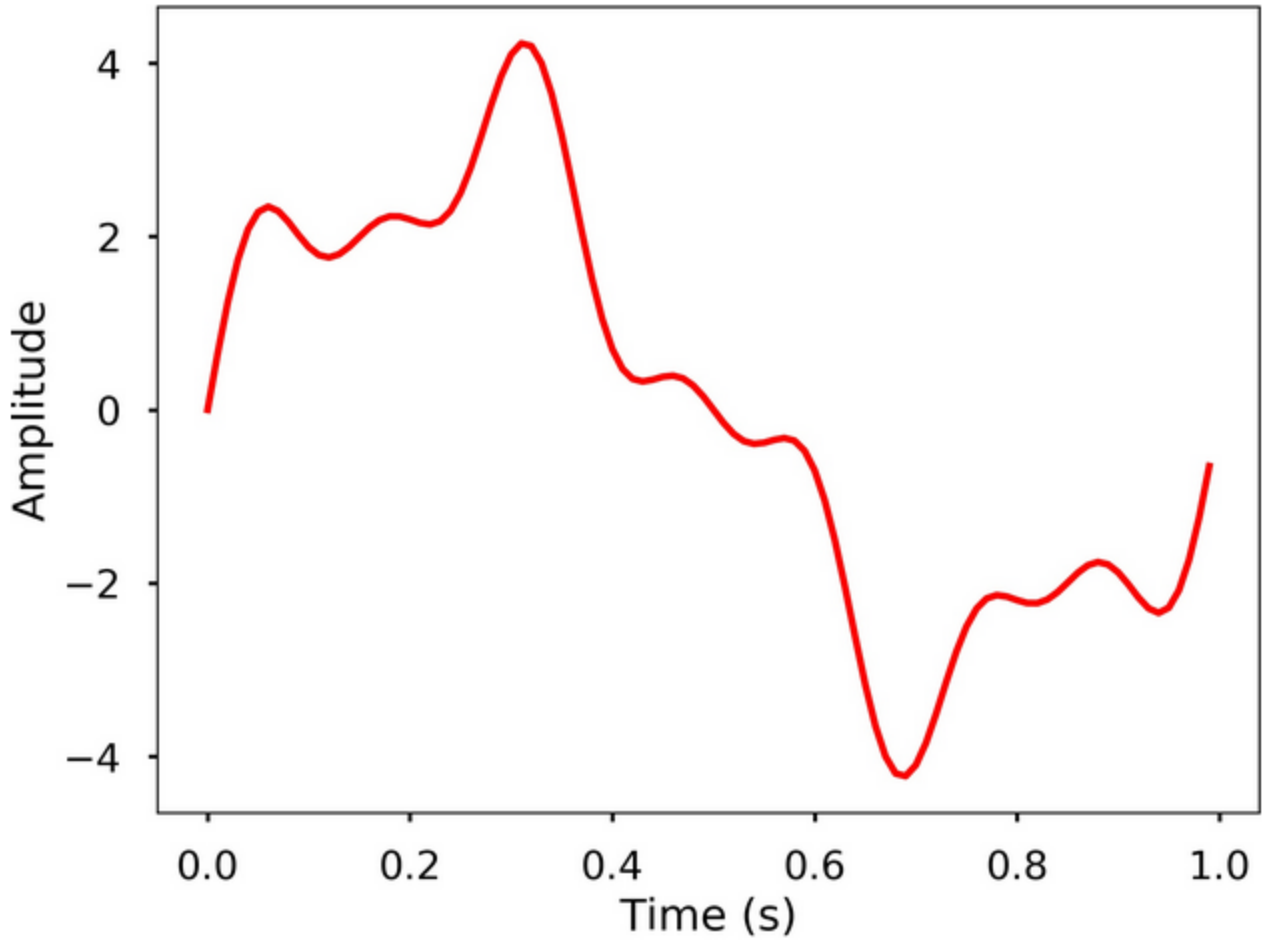
Image: https://en.wikipedia.org/wiki/Fourier_transform



f(x,y) F(*u*,*v*)

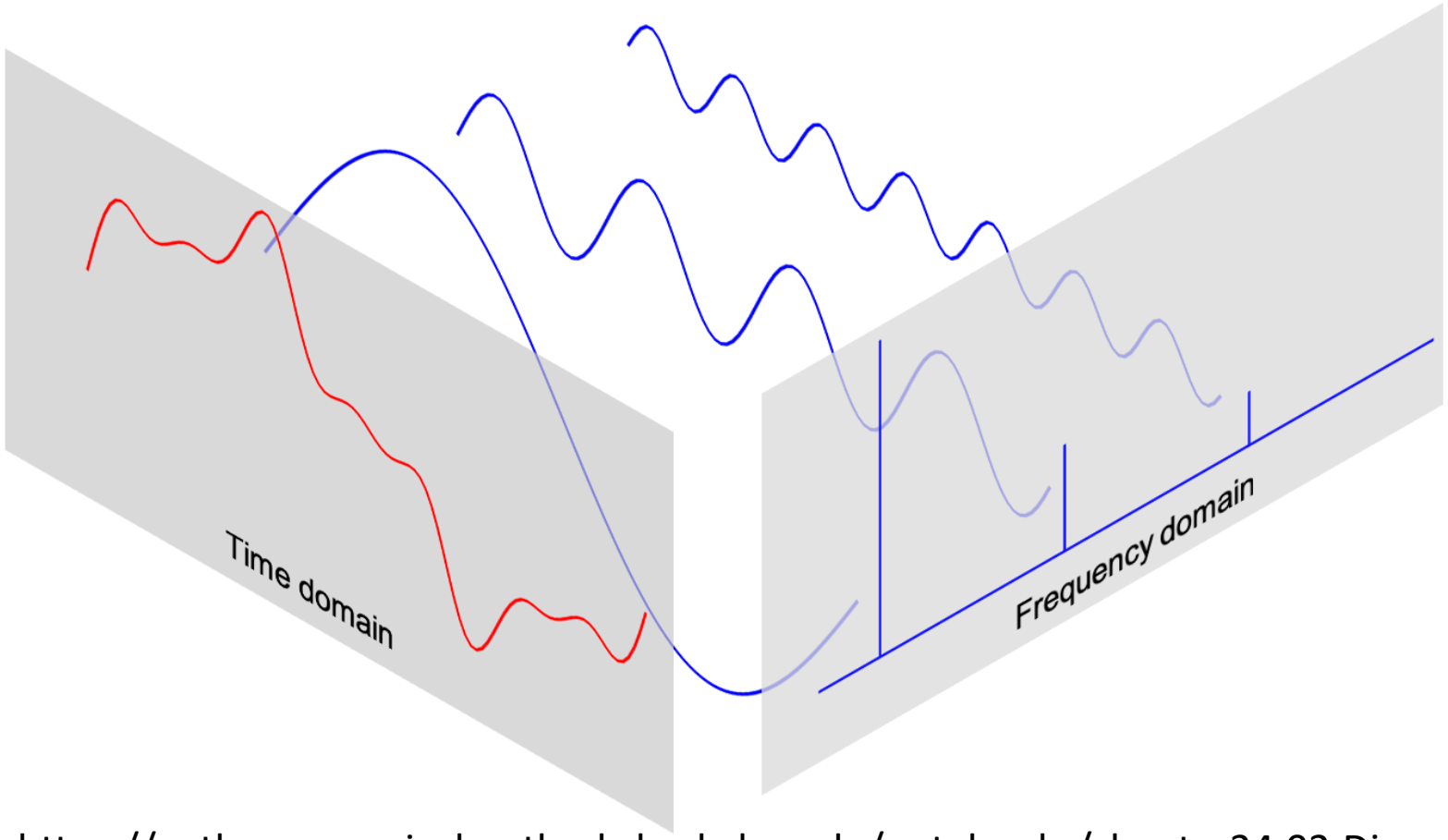
Image:http://measurebiology.org/wiki/Spring_2011:3D_PSF_lab:Kristin

Fourier Transform



https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.02-Discrete-Fourier-Transform.html

Fourier Transform



https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.02-Discrete-Fourier-Transform.html

Discrete Fourier Transform (DFT)

2D DFT

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

$$u = 0, 1, \ldots, M-1 \quad v = 0, 1, \ldots, N-1$$

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

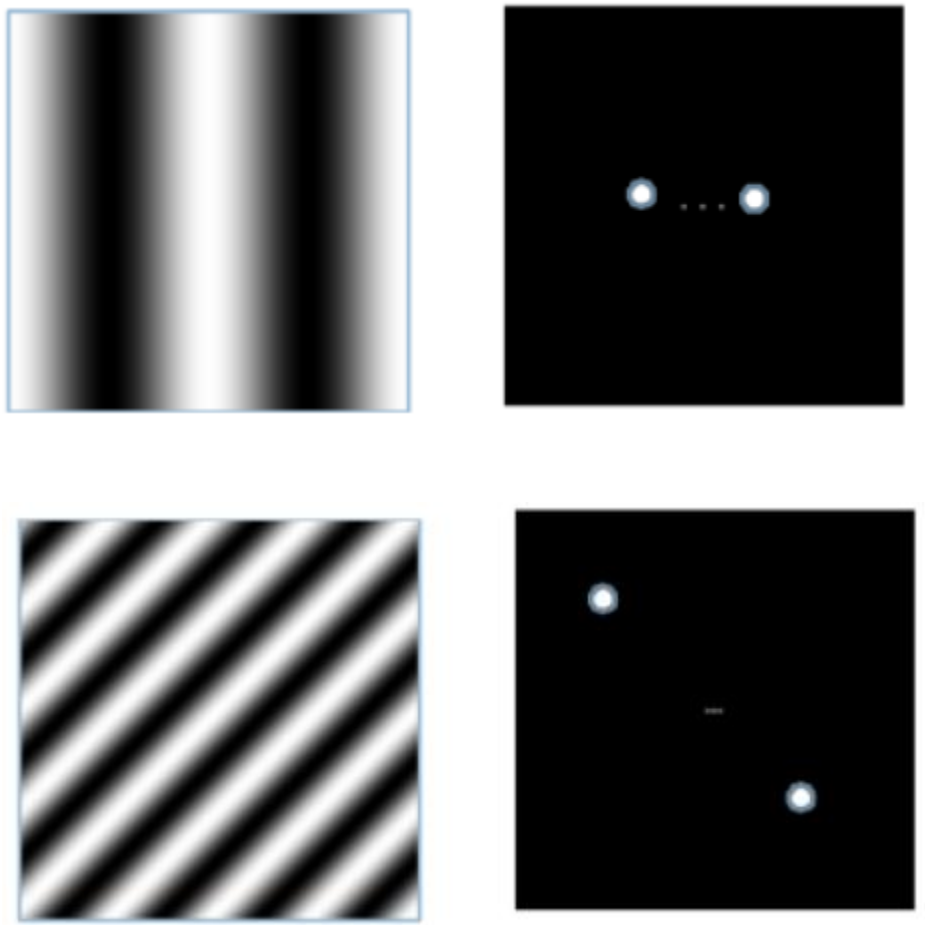
Discrete Fourier Transform (DFT)

2D Inverse DFT

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

$$x = 0, 1, \ldots, M-1 \quad y = 0, 1, \ldots, N-1$$

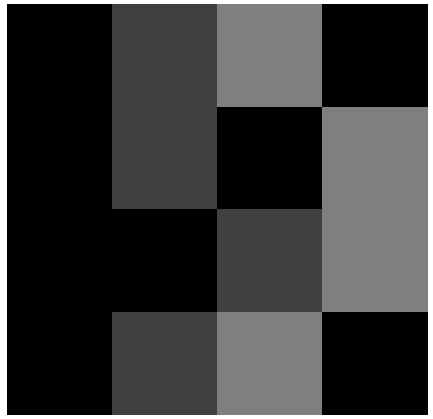
DFT



https://sbme-tutorials.github.io/2018/cv/notes/3_week3.html

DFT Example (4x4 pixel gray image)

For the image shown find Fourier transform at points (0,0), (0,1) and (1,0).

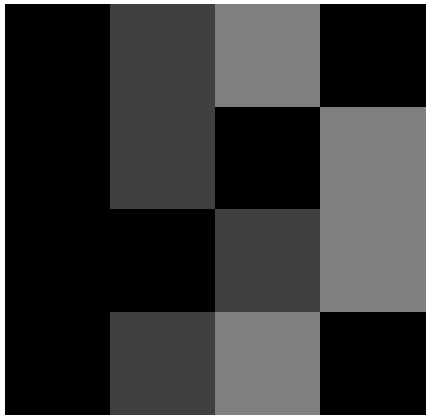


Gray scale: DOI:10.1080/02635143.2012.656084

DFT Example (4x4 pixel gray image)

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

x / y	0	1	2	3
0	0	2	1	0
1	0	2	0	1
2	0	0	2	1
3	0	2	1	0



$$F(0,0) = \sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)ee^{-2j\pi(\frac{x.0}{4}+\frac{y.0}{4})} = 12$$

$$F(0,1) = \sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)ee^{-2j\pi(\frac{y.1}{4})} = 0$$

$$F(1,0) = \sum_{x=0}^{3}\sum_{y=0}^{3} f(x,y)ee^{-2j\pi(\frac{x.1}{4})} = -4-4j$$

DFT Example (4x4 pixel gray image)

y \ x	0	1	2	3
0	0	2	1	0
1	0	2	0	1
2	0	0	2	1
3	0	2	1	0

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

$$F(0,0) = \sum_{x=0}^{3} \sum_{y=0}^{3} f(x,y) e e^{-2j\pi(\frac{x.0}{4}+\frac{y.0}{4})} = 12$$

$$F(0,1) = \sum_{x=0}^{3} \sum_{y=0}^{3} f(x,y) e e^{-2j\pi(\frac{y.1}{4})} = 0$$

$$F(1,0) = \sum_{x=0}^{3} \sum_{y=0}^{3} f(x,y) e e^{-2j\pi(\frac{x.1}{4})} = -4 - 4j$$

Properties of DFT-1

- DFT is a complex representation of an image.
  - ***magnitude*** and ***phase***

Magnitude:

Represents the weight of each frequency component and gives an intuitive information about frequency spectrum of the image.

$$f_{\text{mag}}[m,n] \longleftrightarrow |F[i,k]|$$

Phase:

Retain information about image edges and image orientations.

$$f_{\text{ph}}[m,n] \longleftrightarrow e^{j \arg\{F[i,k]\}}$$

DFT: Magnitude and Phase Components

Original

Magnitude

Phase

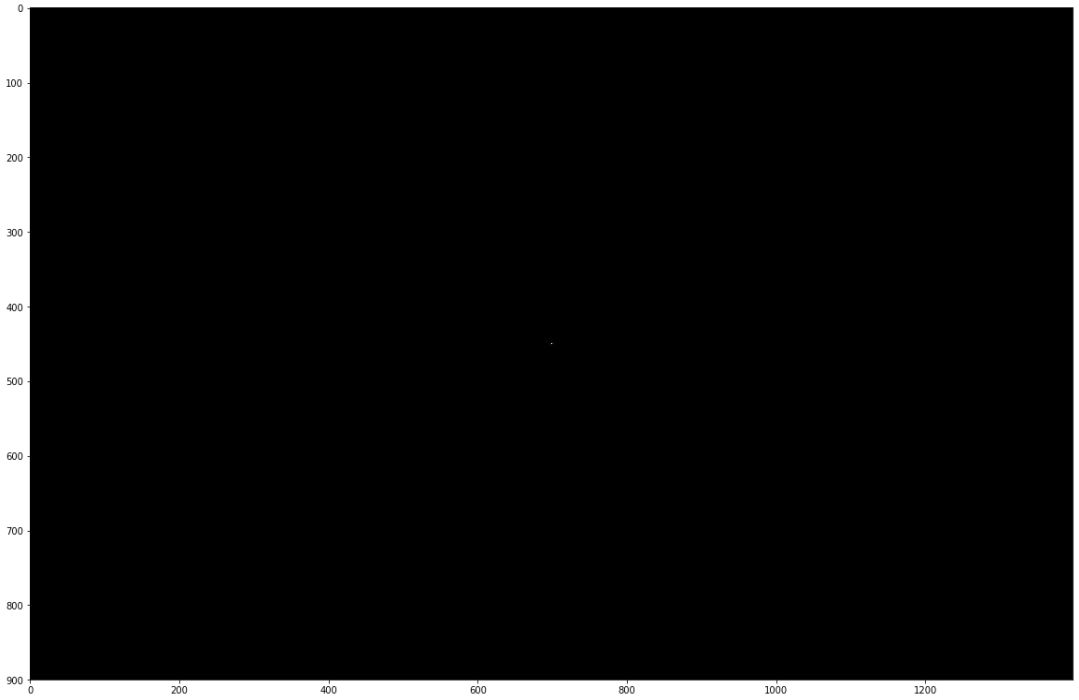
May need further filters….

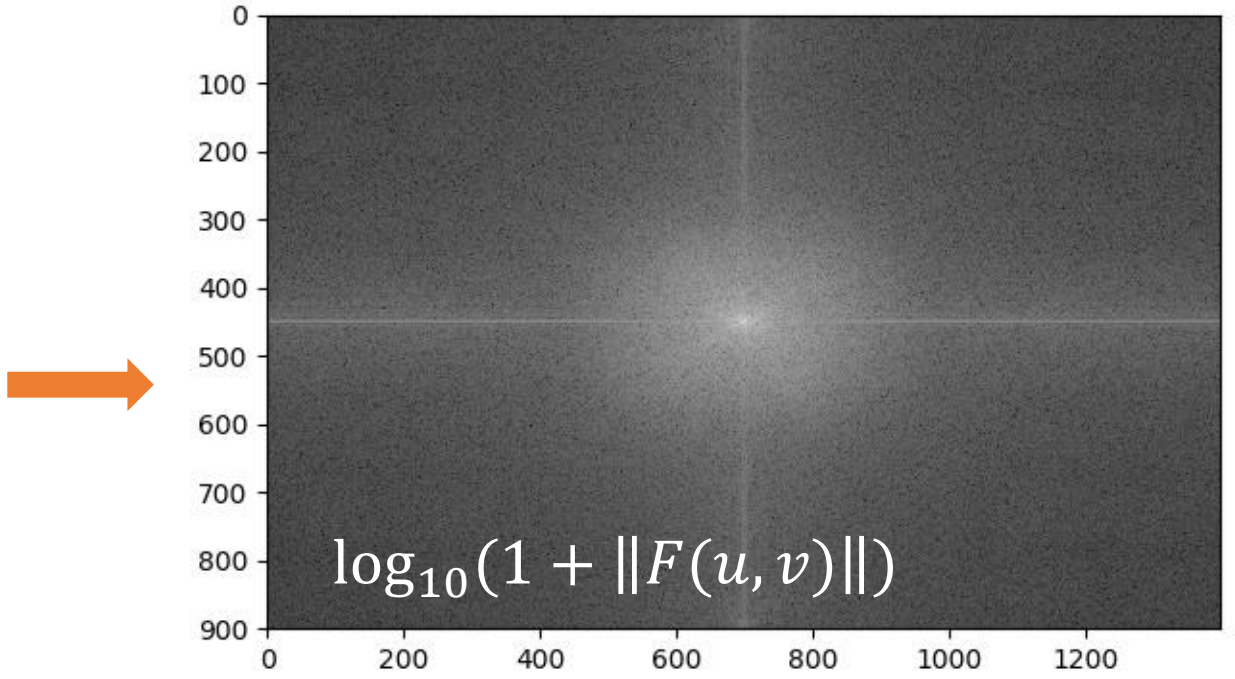
https://diego.assencio.com/?index=81059bc883bd6156c2e83f83e2e38c2b

DFT: Logarithmic Magnitude

Due to large variations in magnitudes usually, magnitudes are displayed in a log scale to compress the dynamic range.

One can only recognize the largest value in the center of the image. All remain values appear as black on the screen.





Properties of DFT-2

- DFT has ***high*** and ***low*** frequency components.
  - Central regions represent the low frequency components
  - Peripheral regions represent the high frequency components
  - Center of the image represents the domain center value with zero frequency (which is the total intensity of the image)

Two views of filtering

- Image filters in spatial domain
  - Filter is a mathematical operation on values of each patch
  - Smoothing, sharpening, measuring texture
- Image filters in the frequency domain
  - Filtering is a way to modify the frequencies of images
  - Denoising, sampling, image compression

Image filtering

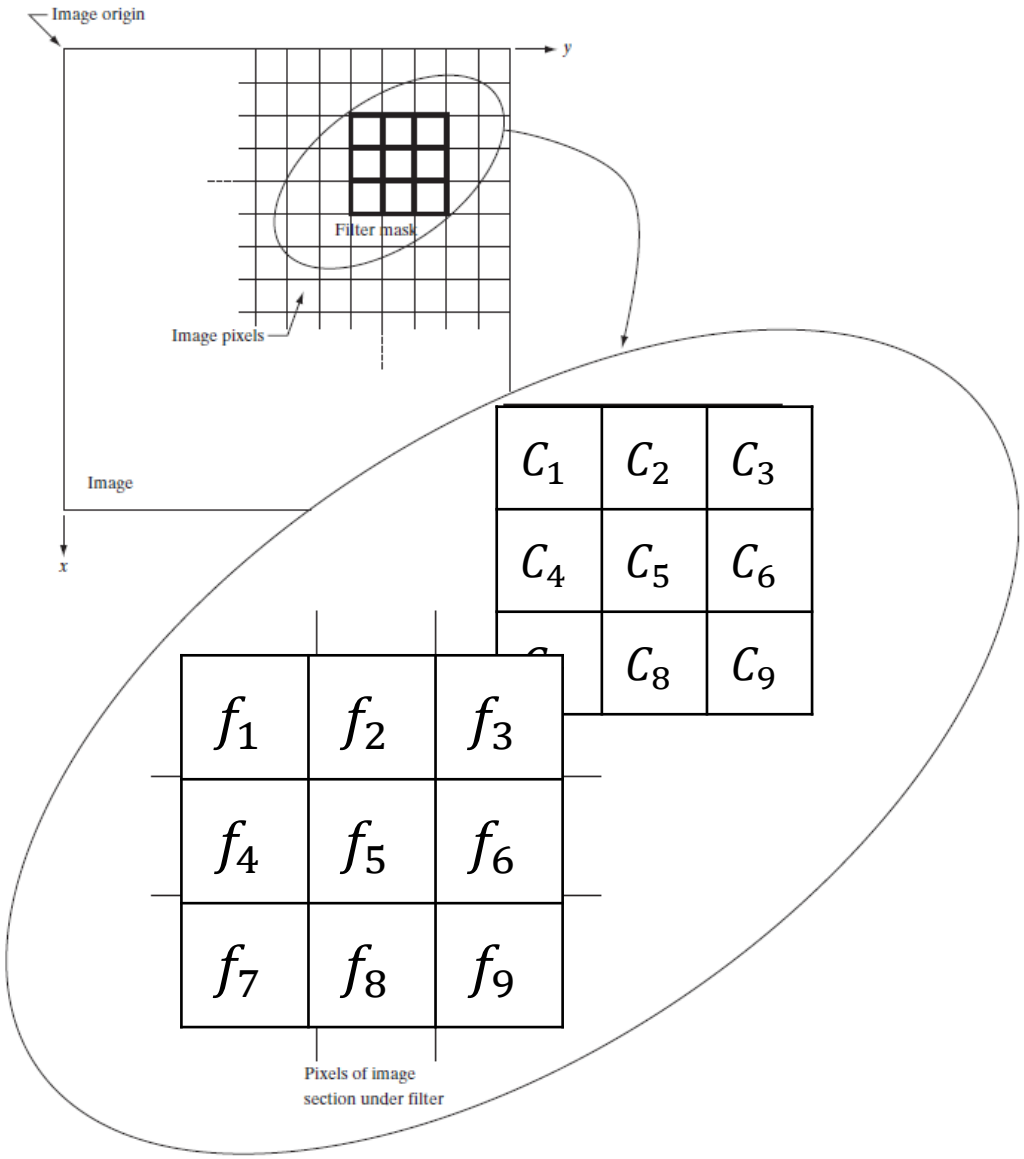
- Image filtering: for each pixel, compute function of local neighborhood and output a new value
  - Same function applied at each position
  - Output and input image are typically the same size

Image filtering

- Linear filtering: function is a weighted sum/difference of pixel values

- Really important!
  - Enhance images
    - Denoise, smooth, increase contrast, etc.
  - Extract information from images
    - Texture, edges, distinctive points, etc.
  - Detect patterns
    - Template matching

Image Filtering with Masks



Move the mask over the image, calculate the pixel value using correlation or convolution

Adapted from Digital Image Processing 3rd Edition
by Rafael Gonzalez (Author), Richard Woods (Author)

1D Correlation and Convolution

Correlation of two functions

$$g(x) = w(x) \circ f(x) = \sum_{s=-a}^{a} w(s) f(x+s)$$

Convolution of two functions (like rotate w for 180 degree)

$$g(x) = w(x) * f(x) = \sum_{s=-a}^{a} w(s) f(x-s)$$

with image of size M and mask size m, a = (m-1)/2, x = 0, …, M-1

2D Correlation and Convolution

Correlation of two functions

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

Convolution of two functions (like rotate w for 180 degree)

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

Linear Filtering of an image f of size MxN filter mask of size mxn is given by the expression, a = (m-1)/2, b = (n-1)/2,
x = 0, 1, 2, … , M-1 and y = 0, 1, 2, … , N-1

Spatial Filter (Mask)

Spatial filter: also called mask/kernel/template or window)

- Consist of a neighbourhood with coefficients on pixels
- Example: masks of odd sizes, e.g. 3x3, 5x5,…

Apply a filter to an image: simply move the filter mask from point to point in an image. At each point (x, y), the response of the filter at that point is calculated using a predefined relationship.

Example: Linear Filter

$$D_a = C_1Z_1 + C_2Z_2 + \cdots + C_9Z_9 = \sum_{i=1}^{m\times n} C_iZ_i$$

C_1	C_2	C_3
C_4	C_5	C_6
C_7	C_8	C_9

Example: Box Filter

$$g[\cdot,\cdot]$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Slide credit: David Lowe (UBC)

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10							

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20						

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30					

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30				

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30				
				?					

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30				
						?			
				50					

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Credit: S. Seitz

Box Filter

What does it do?

- Replaces each pixel with an average of its neighborhood

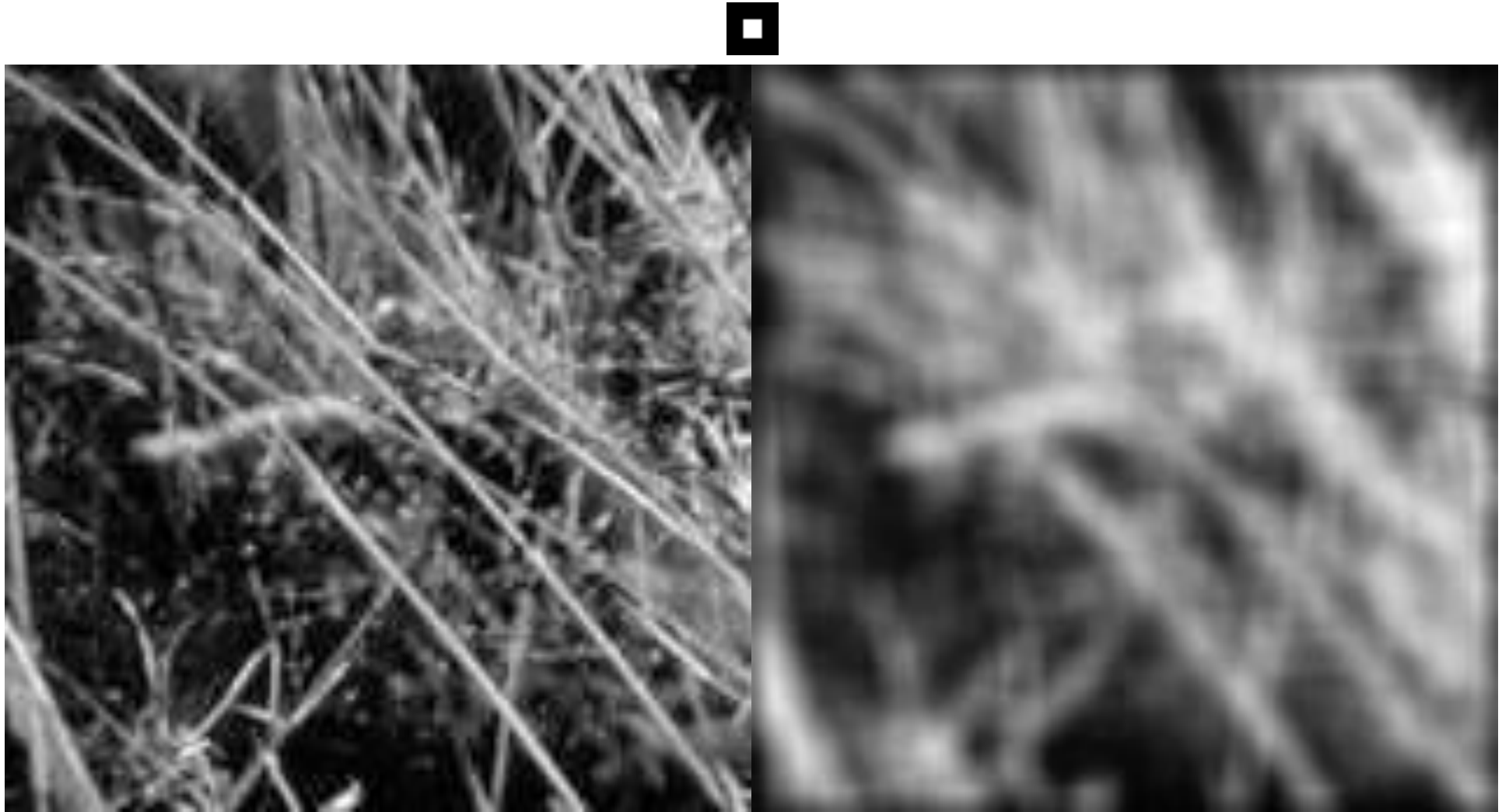
- Achieve smoothing effect (remove sharp features)

$g[\cdot,\cdot]$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Slide credit: David Lowe (UBC)

Smoothing with box filter



Practice with linear filters

Original

0	0	0
0	1	0
0	0	0

?

Source: D. Lowe

Practice with linear filters

Original

0	0	0
0	1	0
0	0	0

Filtered
(no change)

Source: D. Lowe

Practice with linear filters

Original

0	0	0
0	0	1
0	0	0

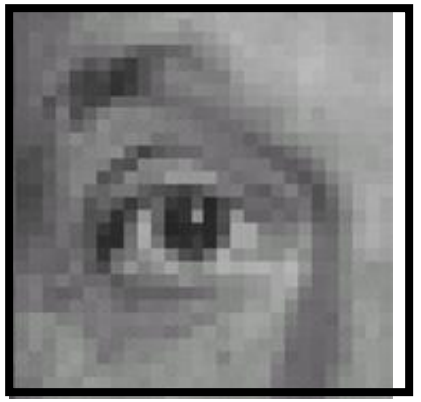
?

Source: D. Lowe

Practice with linear filters

Original

0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

Source: D. Lowe

Practice with linear filters

Original

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad ?$$

(Note that filter sums to 1)

Source: D. Lowe

Practice with linear filters

Original

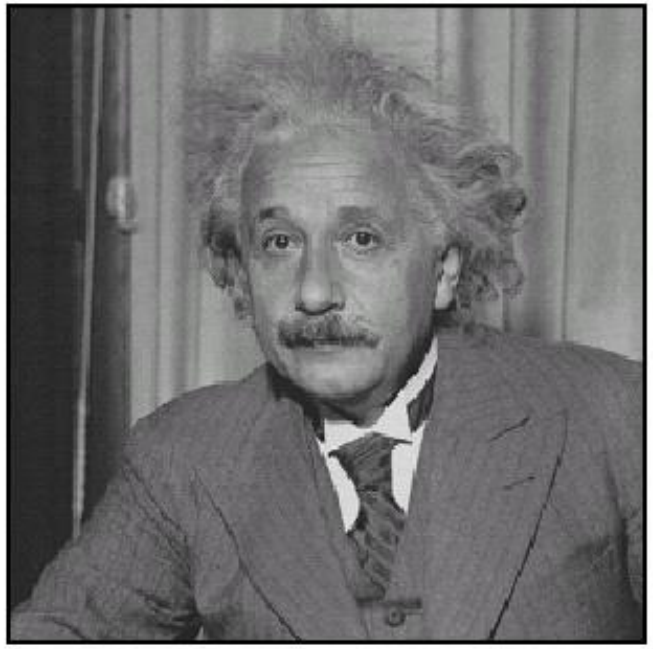
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Sharpening filter

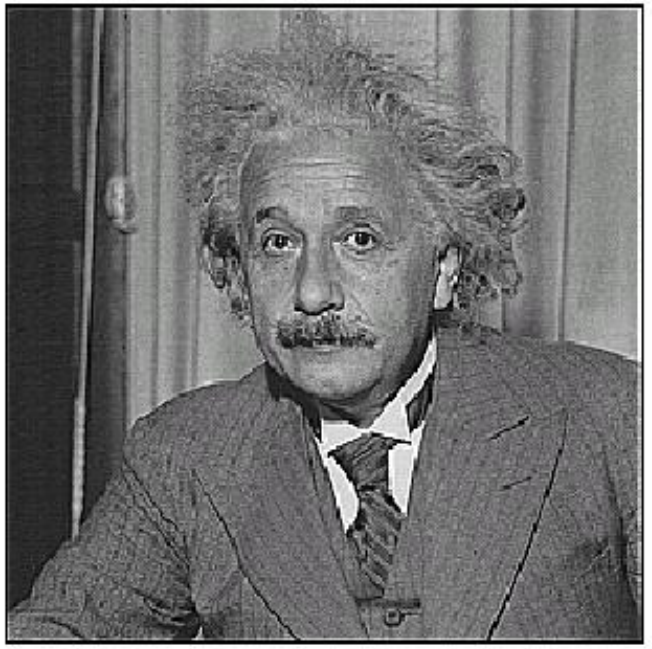
- Accentuates differences with local average

Source: D. Lowe

Sharpening



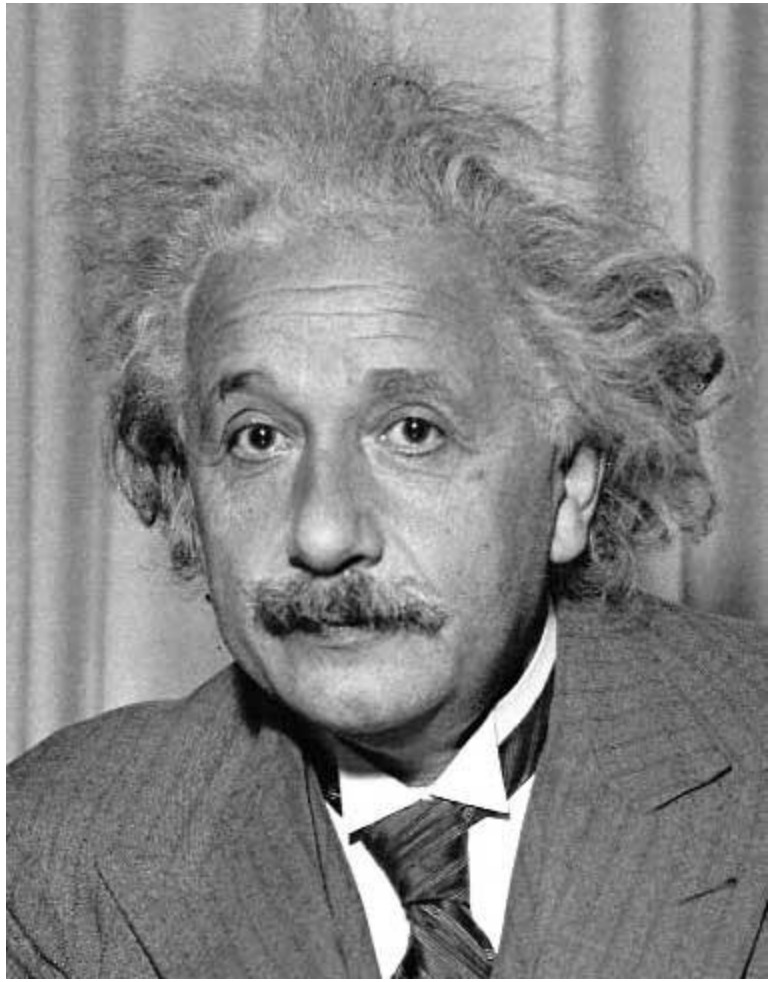
before



after

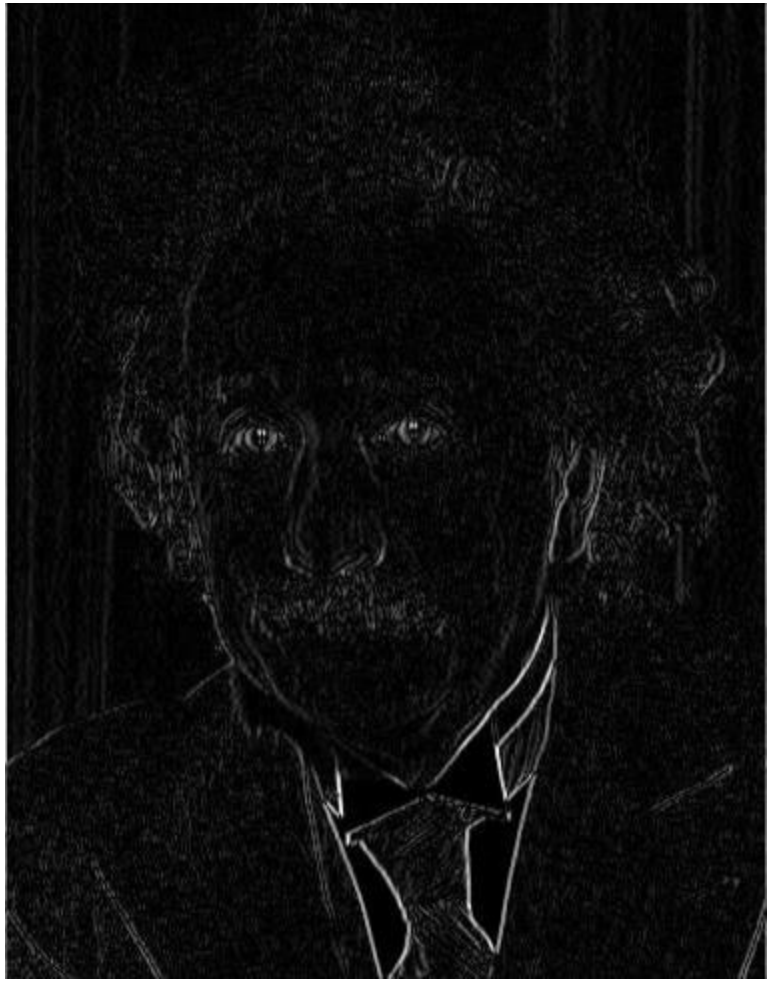
Source: D. Lowe

Other filters



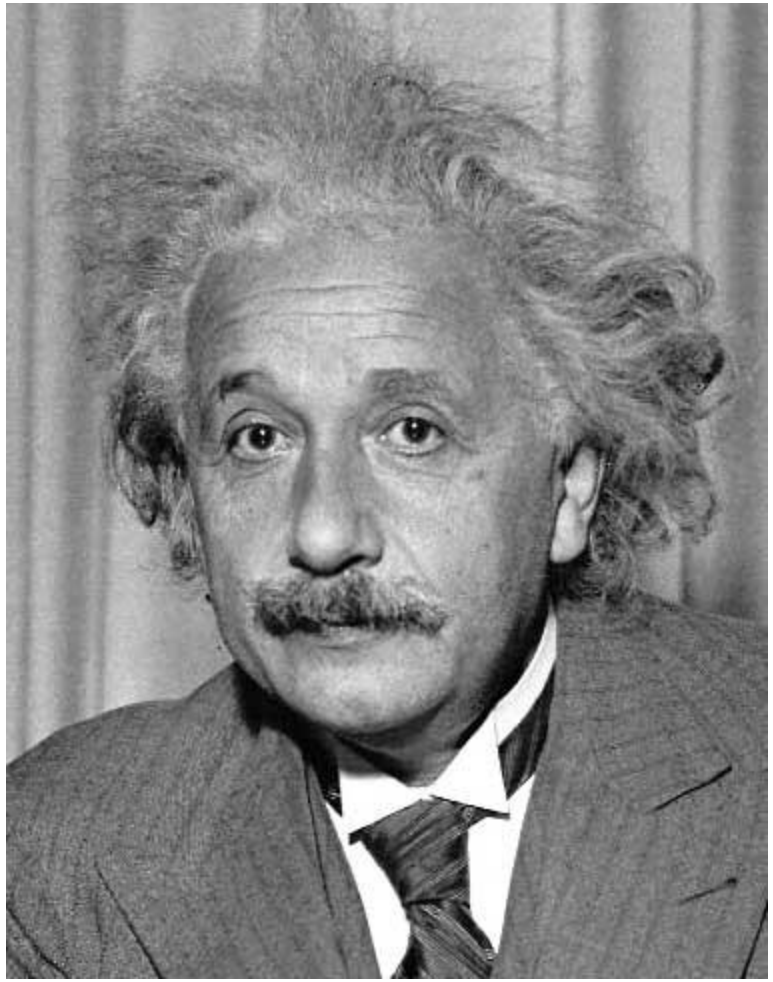
1	0	-1
2	0	-2
1	0	-1

Sobel



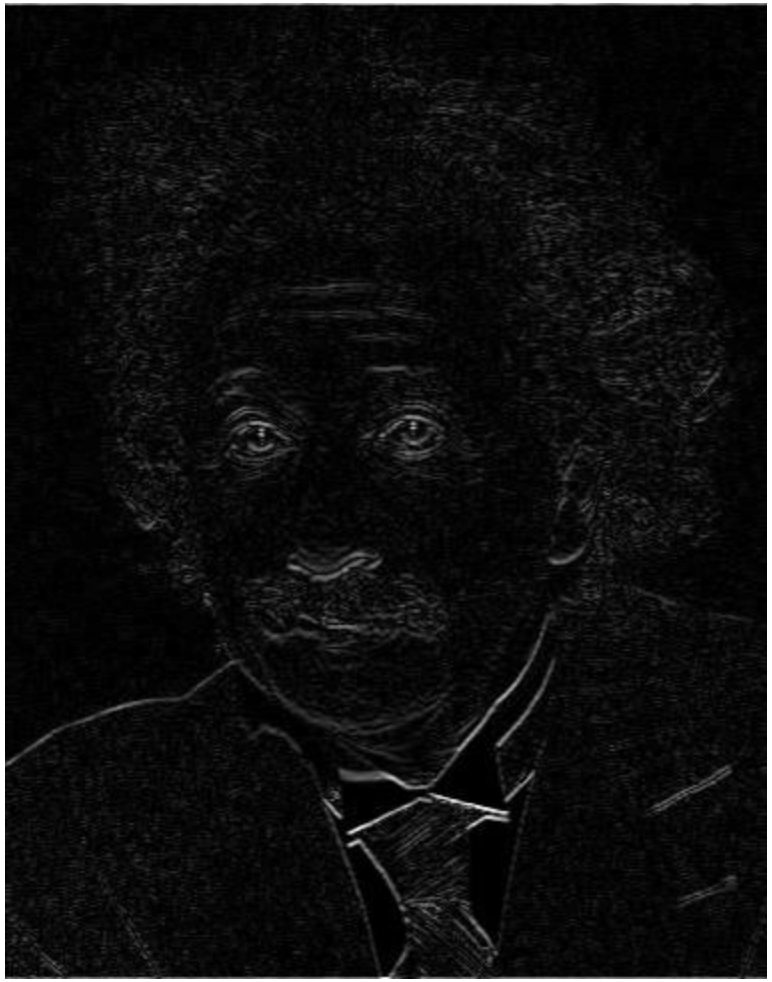
Vertical Edge
(absolute value)

Other filters



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

Sobel



Horizontal Edge
(absolute value)

Basic gradient filters

Horizontal Gradient

0	0	0
-1	0	1
0	0	0

or

-1	0	1

Vertical Gradient

0	-1	0
0	0	0
0	1	0

or

-1
0
1

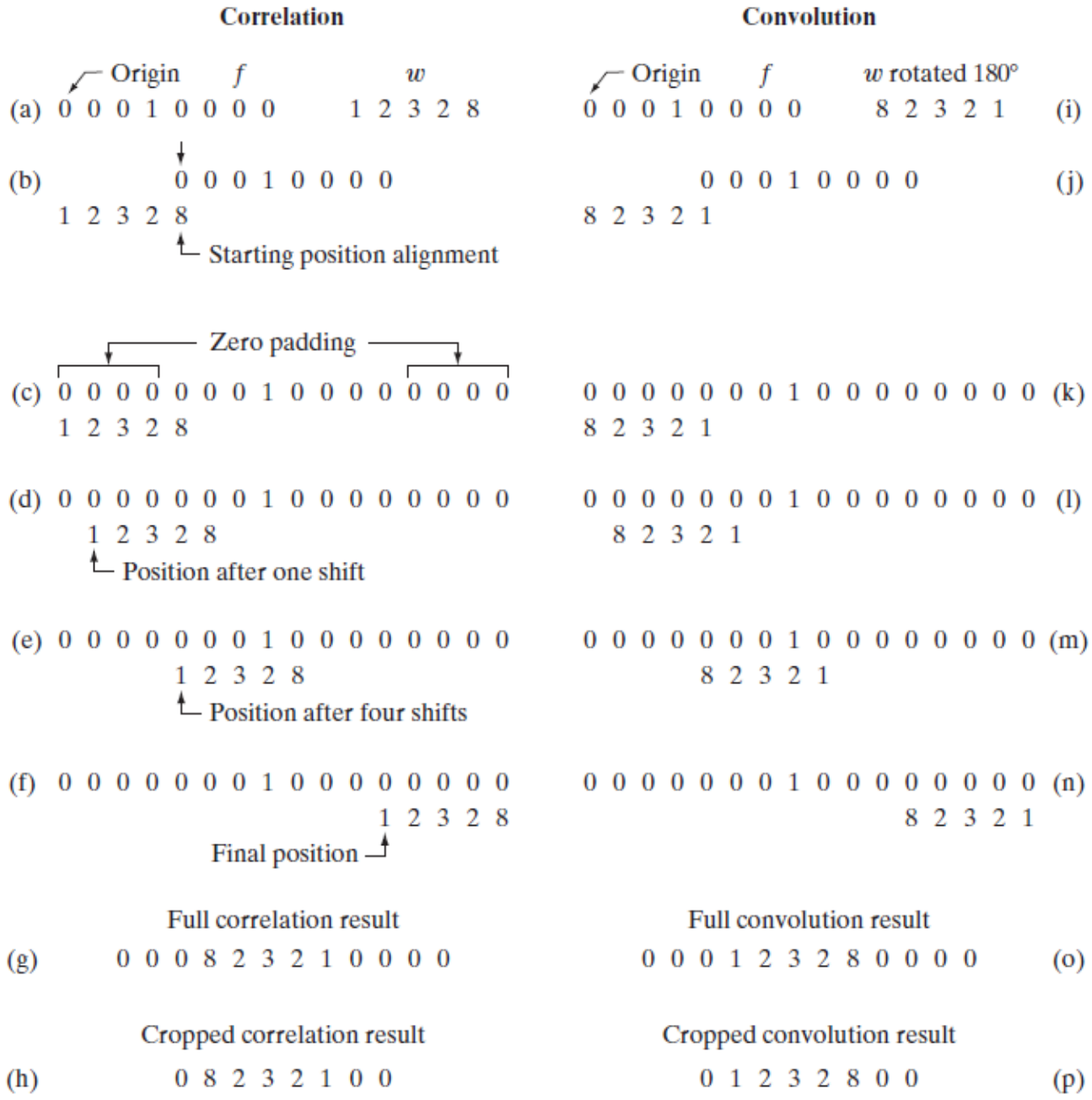


FIGURE 3.29 Illustration of 1-D correlation and convolution of a filter with a discrete unit impulse. Note that correlation and convolution are functions of *displacement*.

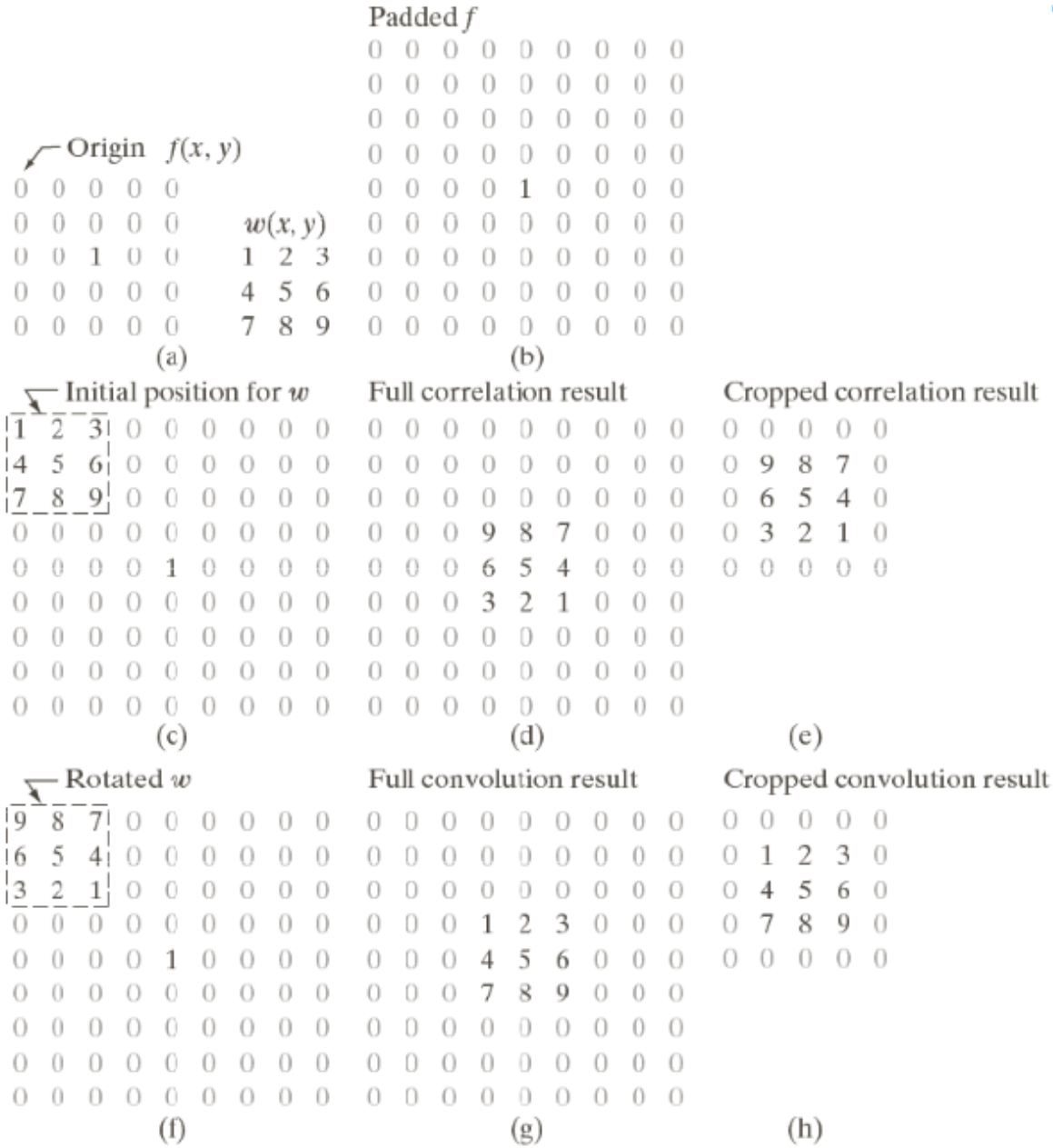


FIGURE 3.30 Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.

Smoothing Spatial Filters

Used for blurring and for noise reduction

Blurring is used in preprocessing steps, such as

- removal of small details from an image prior to object extraction
- bridging of small gaps in lines or curves

Noise reduction can be accomplished by blurring with a linear filter and also by a nonlinear filter

General form of smooth filter (size mxn, m and n are odd)

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

Smoothing Spatial Filters

Output is simply the average of the pixels contained in the neighborhood of the filter mask. Also called averaging filters or low pass filters

Replacing the value of every pixel in an image by the average of the gray levels in the neighborhood will reduce the "sharp" transitions in gray levels.

Sharp transitions:

random noise in the image
edges of objects in the image

Smoothing can reduce noises (desirable) and blur edges (undesirable)

Examples of 3x3 Smoothing Filter

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Box Filter

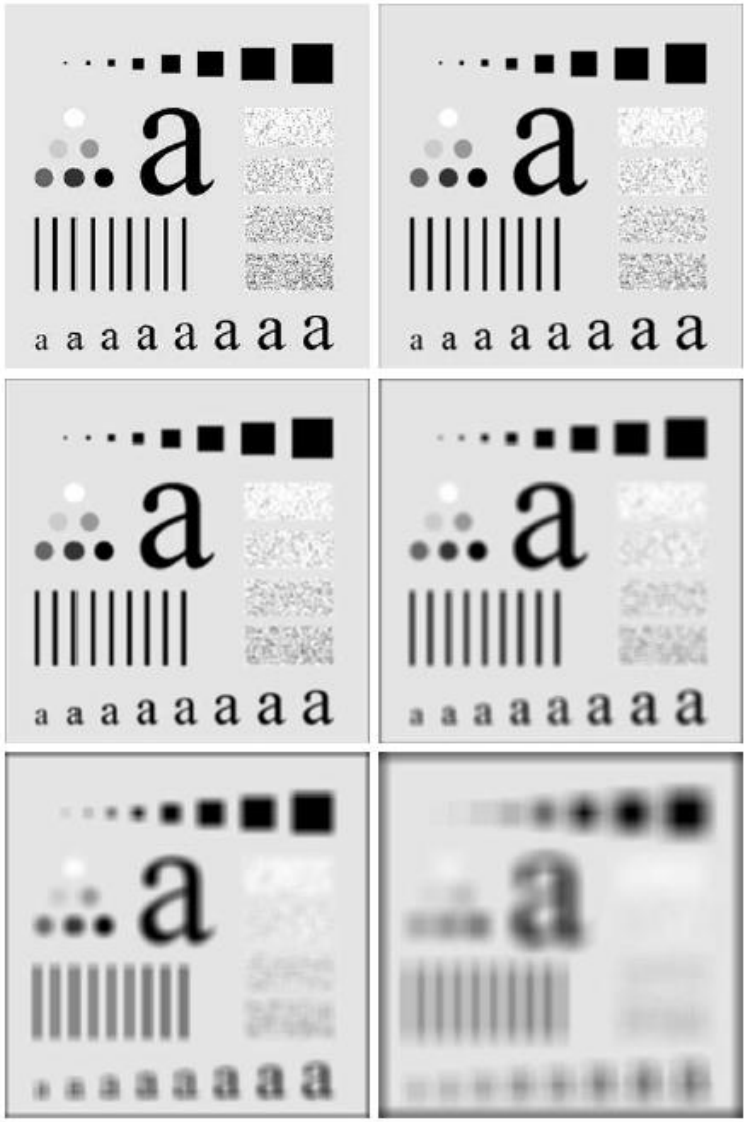
Gives a standard average of the pixels under the mask. (a.k.a low pass filter)

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

Weighted Average

Gives more importance (weight) to some pixels at the expense of others.

Example



- a). original image 500x500 pixel
- b). - f). results of smoothing with square averaging filter masks of size n = 3, 5, 9, 15 and 35, respectively.
- Note:
  - big mask is used to eliminate small objects from an image.
  - the size of the mask establishes the relative size of the objects that will be blended with the background.

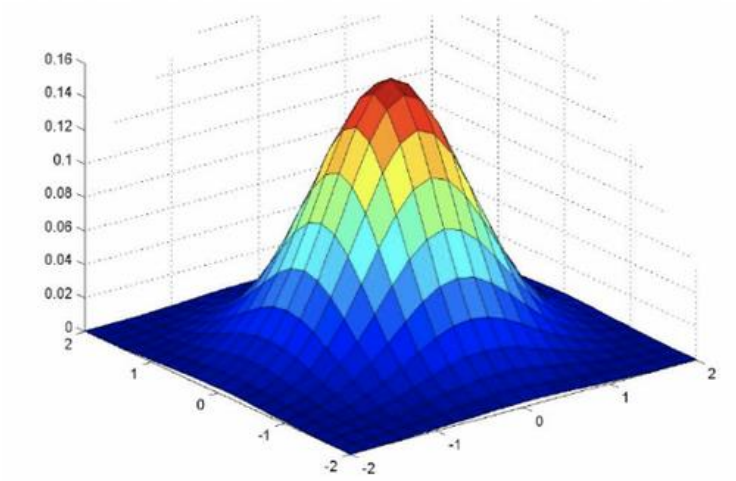
a	b
c	d
e	f

Digital Image Processing 3rd Edition
by Rafael Gonzalez (Author), Richard Woods (Author)

Gaussian Filter

Use Gauss function to generate a mask of m x m matrix *w*

$$h(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$



3x3 matrix

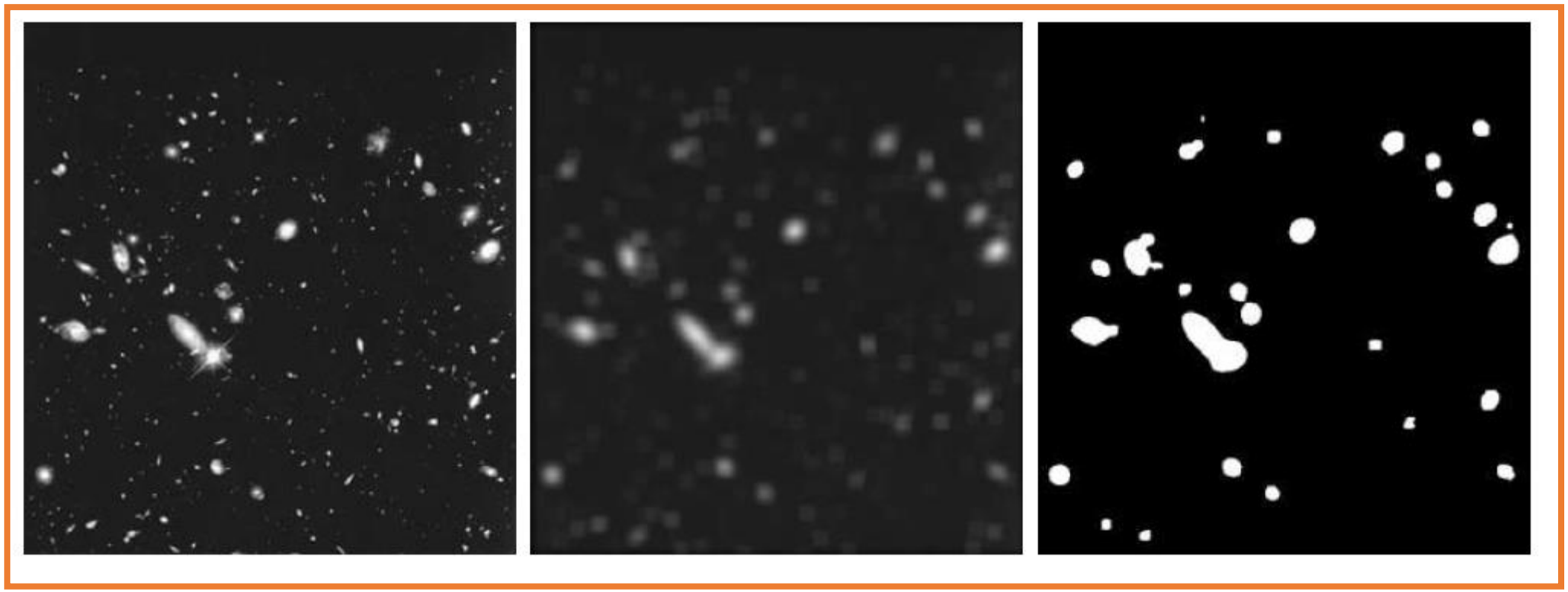
$$w = \begin{bmatrix} h(-1,-1) & h(-1,0) & h(-1,1) \\ h(0,-1) & h(0,0) & h(0,1) \\ h(1,-1) & h(1,0) & h(1,1) \end{bmatrix}$$

Example

Original

Filtered with a
15 x 15 averaging mask

Filtered with a
threshold



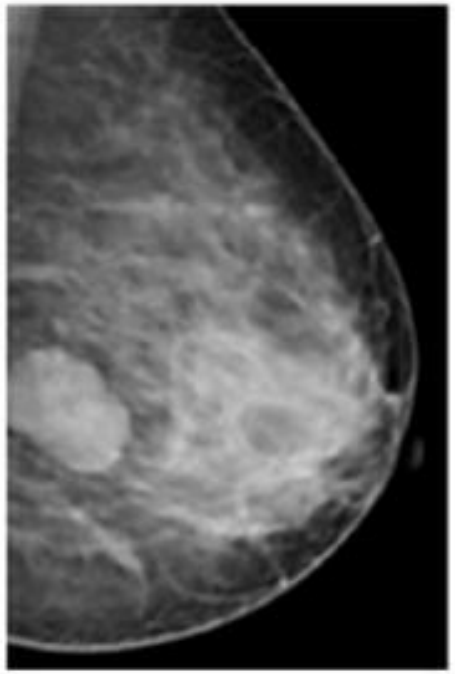
Digital Image Processing 3rd Edition
by Rafael Gonzalez (Author), Richard Woods (Author)

Low Pass Filter

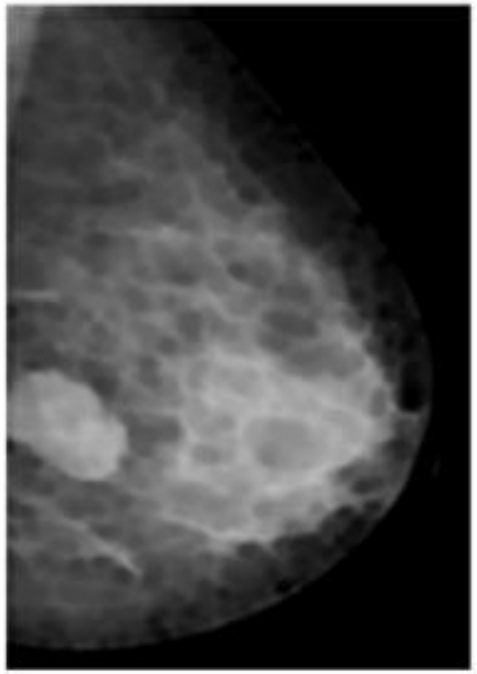
Used to pass low frequency details and remove high frequency components such as sharp transitions or edges.

1	1	1
1	1	1
1	1	1

Original



Filtered (LPF)



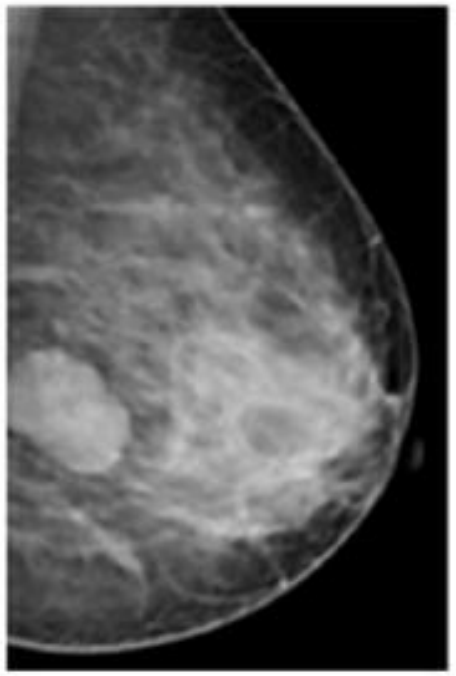
Patel, G.R.. Medical Image Processing: Concepts and Applications

High Pass Filter (HPF)

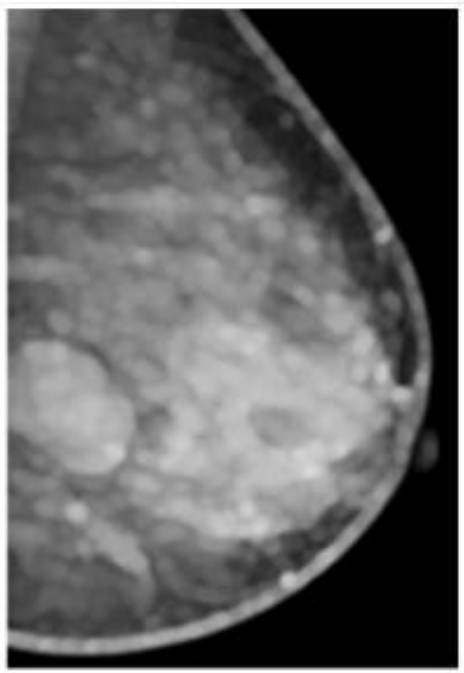
Used to pass high frequency components in spatial domain and reject or attenuate low frequency details of an image. The calculation of filter coefficients is done by selecting the value of a center pixel and subtracting it from it half the average value of the surrounding pixels.

–1	–1	–1
–1	–8	–1
–1	–1	–1

Original



Filtered (HPF)



Patel, G.R.. Medical Image Processing: Concepts and Applications

High Boost Filter (HBF)

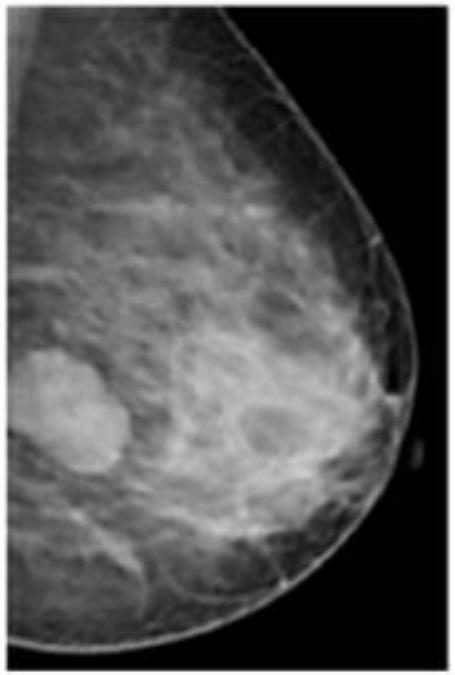
Used to enhance (***boost***) the high frequency content along with the low frequency information of the background.

HBF = (A – 1) Original image + HPF output = A[Original image] – LPF output

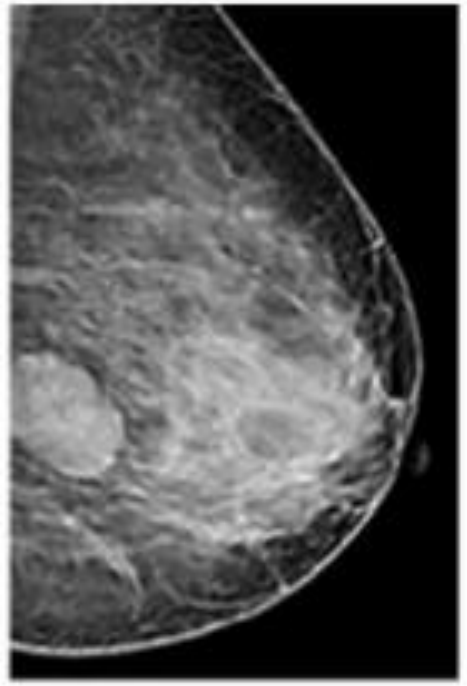
A: High boost coefficient

–1	–1	–1
–1	9 – A	–1
–1	–1	–1

Original



Filtered (HBF)



Patel, G.R.. Medical Image Processing: Concepts and Applications

Frost Filter (FF)

Adaptive and exponentially weighted averaging filter based on the variation coefficient (**dumping factor**). Used to remove speckles.

The filter replaces the pixel of consideration with a weighted sum of the values within the moving kernel of a suitable size.

$$I_s = \sum_{p \in \eta_s} m_p I_p$$

$$m_p = exp\left(-KC_s^2 d_{s,p}\right) / \sum exp\left(-KC_s^2 d_{s,p}\right)$$

$$d_{s,p} = \sqrt{(x - x_p)^2 + (y - y_p)^2)}$$

K: dumping factor
I_s: intensity value of the center pixel in kernel
C_s: local statistic value used for adaptive computation of the filter coefficients
(x,y) and (x_p,y_p) indicate grid coordinates of the centre of the window and the pixel p, respectively

https://www.mdpi.com/1424-8220/19/13/2903/htm

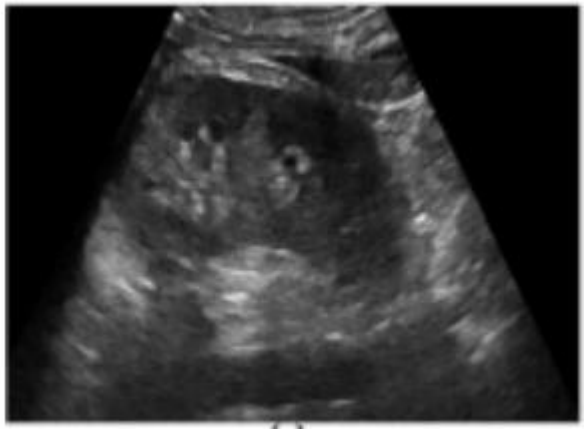
Patel, G.R.. Medical Image Processing: Concepts and Applications

Frost Filter (FF)

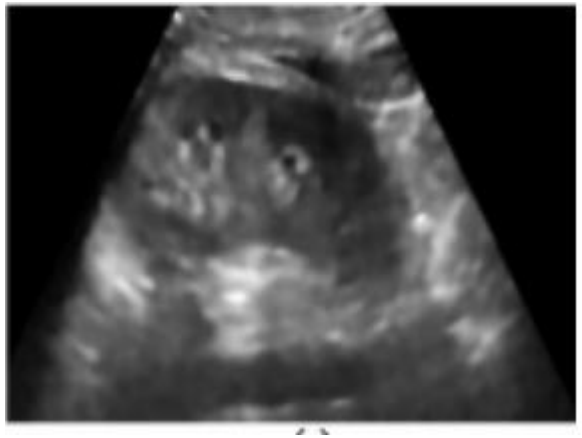
Adaptive and exponentially weighted averaging filter based on the variation coefficient (**dumping factor**). Used to remove speckles.

The filter replaces the pixel of consideration with a weighted sum of the values within the moving kernel of a suitable size.

Original



Filtered (FF)

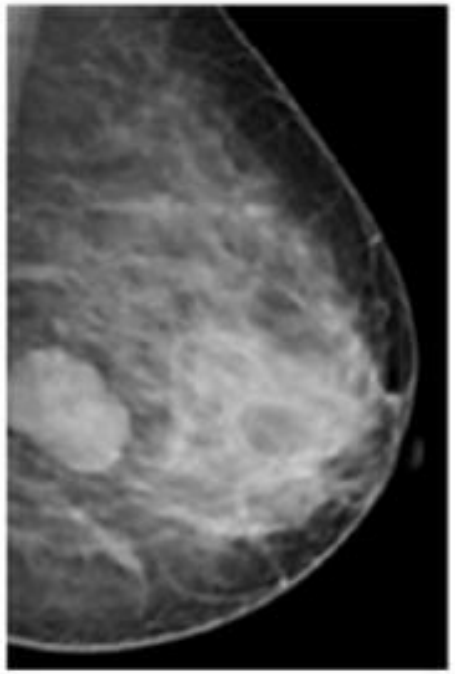


Patel, G.R.. Medical Image Processing: Concepts and Applications

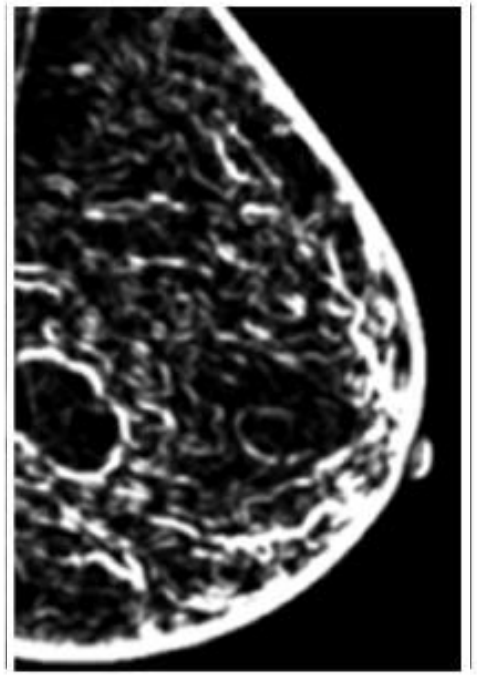
Variance Filter (VF)

A variance filter is used to highlight the edges in the X-ray and CT scan images of the brain. The areas of varying reflectance become brighter than the uniform areas that appear darker.

Original



Filtered (VF)



Patel, G.R.. Medical Image Processing: Concepts and Applications

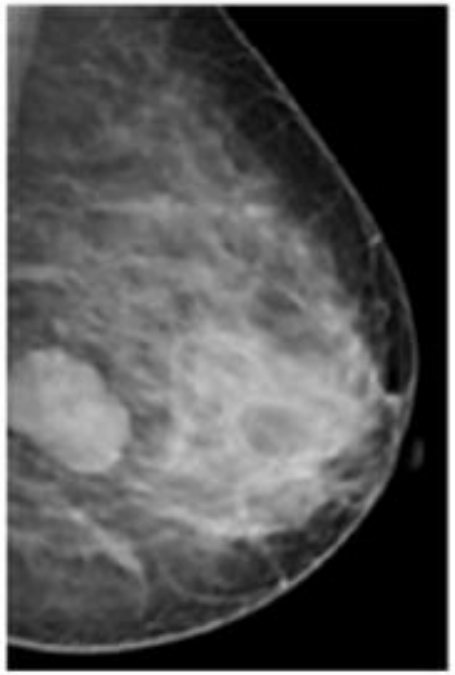
Median Filter (MF)

The median value of the pixels in the window is computed and then, the center pixel of the window is replaced with the computed median value.

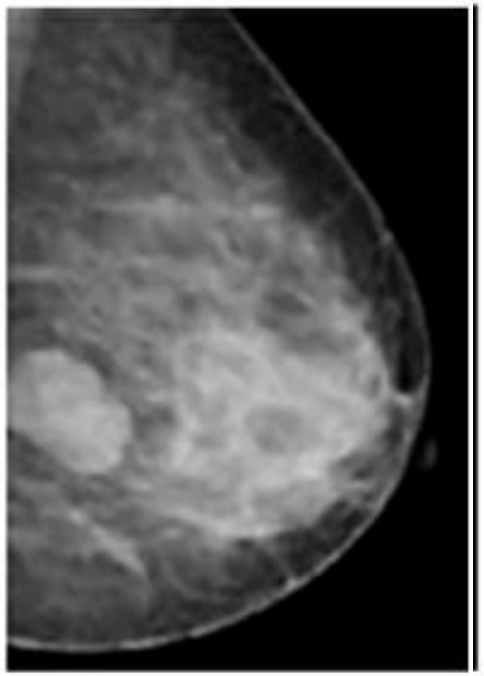
Can substitute LPF, since LPFs are not suitable for reducing the strong spike-type noise pattern.

25	16	18
32	40	37
54	47	51

Original



Filtered (MF)



Patel, G.R.. Medical Image Processing: Concepts and Applications

Which filters to use?

Steps for Analysis

- Use images from the same imaging device and enhance them using different spatial filters. Discussed.
- First, add a uniformly distributed multiplicative noise with a mean (zero) and variance (e.g., 0.04) to the image.
- Then analyze the performance of filters by using some metrics, e.g., Mean square error (MSE) signal-to-noise ratio (SNR) between the original images and the enhanced images.

Example Results

Table 3.1 Comparison of Filter Performance in terms of MSE and SNR

S. No.	*Filter Type*	*MSE*	*SNR* (dB)
1	Low pass filter	0.785	19.212
2	High pass filter	0.192	15.314
3	High boost filter	0.262	20.976
4	Frost filter	0.138	22.759
5	Variance filter	0.795	9.156
6	Median filter	0.762	18.232

Patel, G.R.. Medical Image Processing: Concepts and Applications

Which filters to use?

Example Interpretations

- SNR improvement indicates enhanced signal strength and reduced noise level.
- MSE should be small for a good filter. The loss of contrast due to speckle noise affects the texture of images also.
- High boost and frost filters have better results.
- The variance filter is very simple to implement but being non-adaptive, it can deteriorate the performance.

Example Results

Table 3.1 Comparison of Filter Performance in terms of MSE and SNR

S. No.	*Filter Type*	*MSE*	*SNR* (dB)
1	Low pass filter	0.785	19.212
2	High pass filter	0.192	15.314
3	High boost filter	0.262	20.976
4	Frost filter	0.138	22.759
5	Variance filter	0.795	9.156
6	Median filter	0.762	18.232

Patel, G.R.. Medical Image Processing: Concepts and Applications

Reading Assignment

https://zenodo.org/record/2783366/files/%2814-22%29Effect%20of%20Frequency%20Domain.pdf

MAT JOURNALS

Journal of Image Processing and Artificial Intelligence
e-ISSN: 2581-3803
Volume 5 Issue 2

Effect of Frequency Domain Filters on Bio-Medical Images

***Suneet Gupta*[1*], *Sarvottam Dixit*[2]**
[1]*Research Scholar,* [2]*Professor*
[1,2]*Department of Computer Science and Engineering, Mewar University, Chittorgarh, Rajasthan, India*
Email: *suneetsonu@yahoo.com*
DOI:

Abstract

This study tries to find out the most effective frequency domain filter used for enhancing biomedical images. Five different biomedical images have been taken and they are filtered with different low and high pass filters at different cut off frequencies. As far as enhancement method is concerned, the method is very simple because the motive here is to find the most effective filter not a good enhancement technique. To assess the enhancement the metric MSE is used. The experiments are done on MATLAB.

Keywords: *Butterworth, frequency domain, Gaussian, Ideal, MSE.*