

Computer Vision, and Digital Image Processing

Prepared by Dr. Mohamed Mostafa

AI 612

Lecture 6



Filtering in the Frequency Domain Part 1

Lecture 6

AI 612

Prepared by
Dr. Mohamed Mostafa

Chapter
04

Spatial Filtering

Resources for this
lecture

Chapter 4:

Gonzalez, Rafael C. *Digital image processing*.
Pearson education, India, 2009.



Image Transformation

Spatial Domain

images or are directly manipulated in terms of their spatial coordinates, pixel values, or neighborhood operations.

Filtering in the spatial domain involves directly modifying the pixel values of the image or signal using spatial convolution operations with a filter kernel or mask.

can preserve edges better since it directly operates on the pixel values and their spatial relationships.

Frequency Domain

images or are represented in terms of their frequency components using techniques like Fourier transforms.

Filtering in the frequency domain involves modifying the frequency components of the image or signal by applying frequency domain filters such as low-pass, high-pass, or band-pass filters.

Frequency domain filtering can be computationally intensive due to the need for Fourier transforms and inverse Fourier transforms. However, once in the frequency domain, certain operations like convolution can be more efficient.

Application of both image transformation Techniques

- **Frequency Domain:** Frequency domain filtering is commonly used in applications where global frequency characteristics of the image or signal are important, such as image compression or spectral analysis.
- **Spatial Domain:** Spatial domain filtering is widely used in applications where local spatial relationships and pixel values are crucial, such as image enhancement or feature extraction.

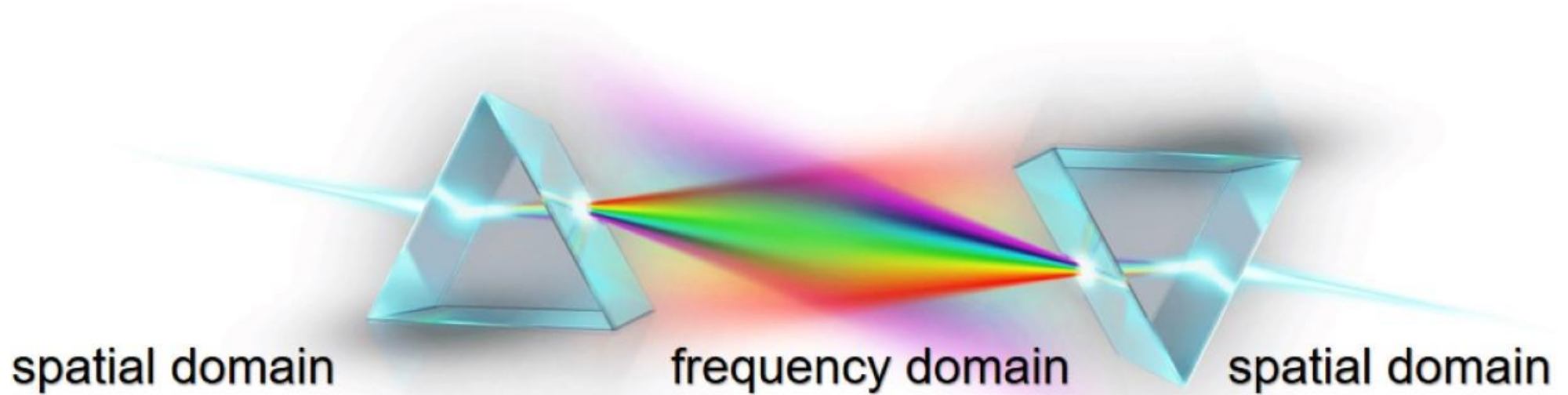
Image Transform meaning

From spatial domain  the frequency domain

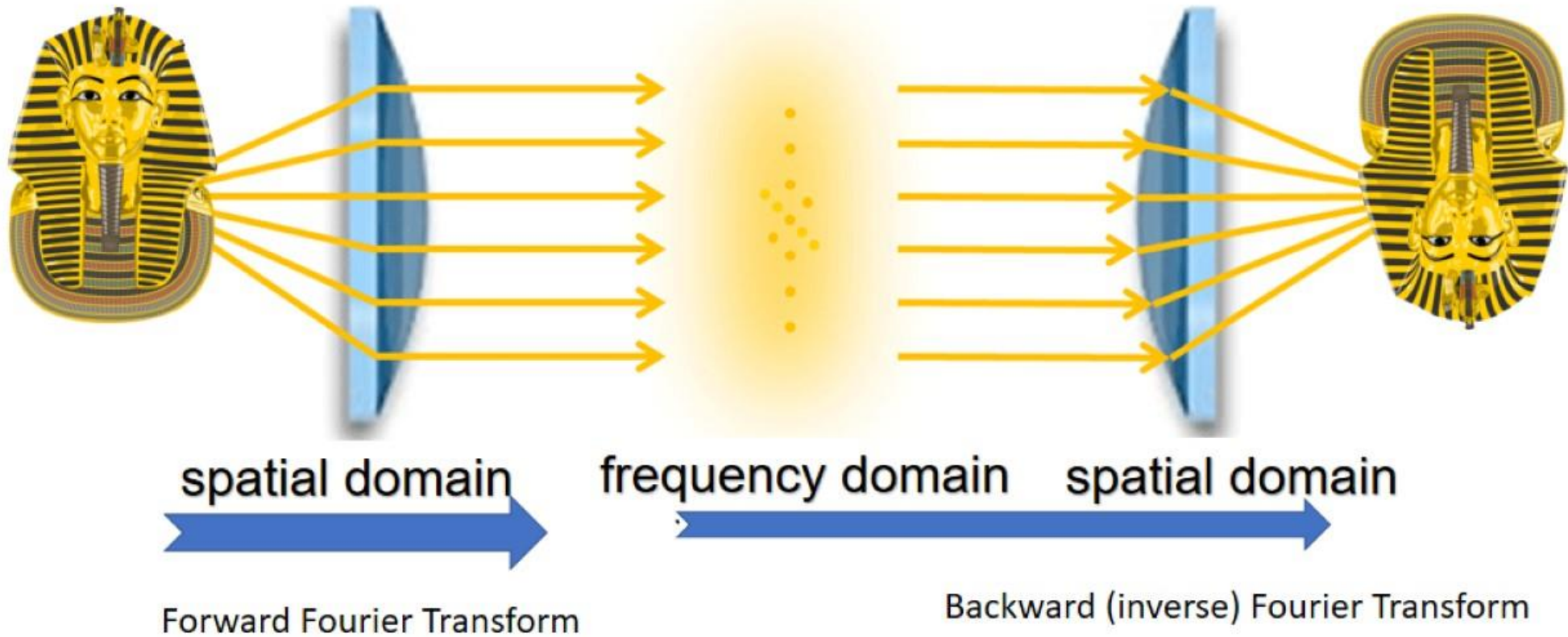
Pixels

A sum of sinusoidal
functions of
different
frequencies and
amplitudes

Frequency Transform



Frequency Transform



Types of Noises

In image processing, various types of noise can corrupt an image, affecting its quality and making it challenging to extract useful information.

- **Gaussian Noise:**
- Gaussian noise is characterized by a Gaussian probability distribution and typically manifests as random variations in pixel intensity.
- In the frequency domain, Gaussian noise appears as high-frequency components scattered throughout the spectrum.
- To mitigate Gaussian noise, frequency domain filtering techniques such as low-pass filtering can be applied to suppress high-frequency components while preserving low-frequency information.



Types of Noises

In image processing, various types of noise can corrupt an image, affecting its quality and making it challenging to extract useful information.

- **Salt and Pepper Noise:**

- Salt and pepper noise introduces random occurrences of very bright or very dark pixels in an image.
- In the frequency domain, salt and pepper noise can appear as spikes or outliers in the frequency spectrum.
- Filtering techniques like median filtering in the frequency domain can help remove these outliers effectively while preserving image details.

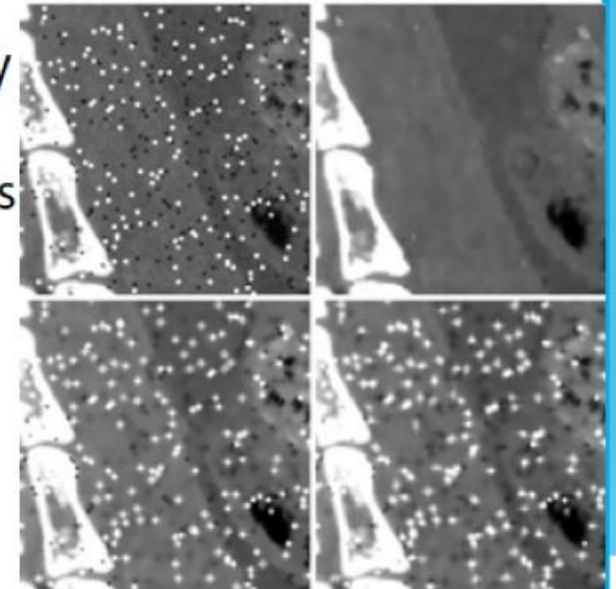


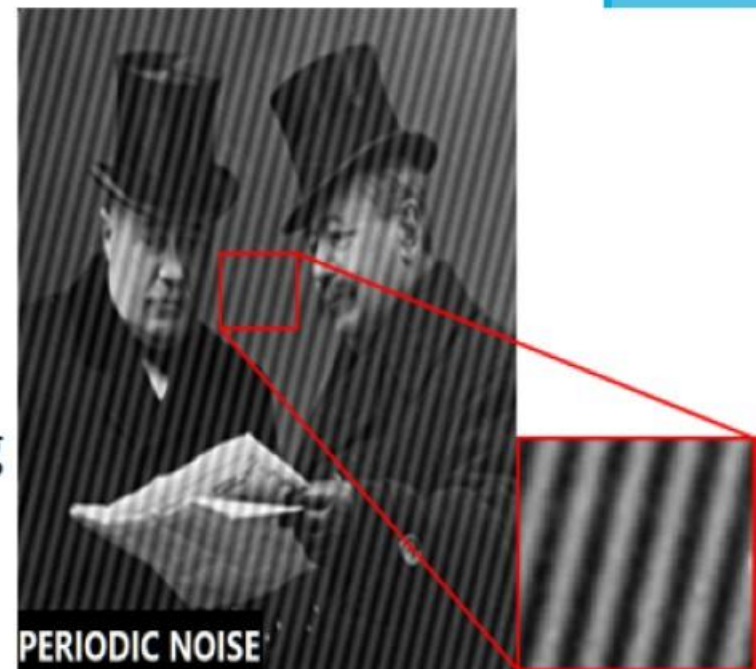
Image source: Michel-González, Eric, Min Hyung Cho, and Soo Yeol Lee. "Geometric nonlinear diffusion filter and its application to X-ray imaging." *Biomedical engineering online* 10, no. 1 (2011): 1-16.

Types of Noises

In image processing, various types of noise can corrupt an image, affecting its quality and making it challenging to extract useful information.

- **Periodic Noise:**

- Periodic noise exhibits regular patterns or periodic variations in pixel intensity across the image.
- In the frequency domain, periodic noise manifests as peaks or lines in specific frequency bands.
- Techniques like notch filtering or band-stop filtering can be employed in the frequency domain to suppress the frequencies corresponding to the periodic noise components.

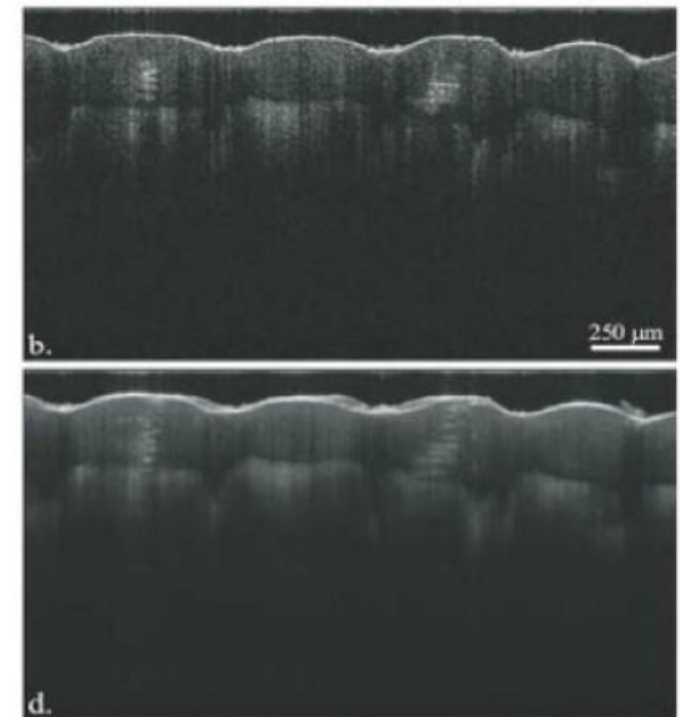


Types of Noises

In image processing, various types of noise can corrupt an image, affecting its quality and making it challenging to extract useful information.

- **Speckle Noise:**

- Speckle noise is commonly found in images acquired through imaging techniques such as ultrasound or synthetic aperture radar (SAR).
- In the frequency domain, speckle noise appears as random variations across different frequency bands.
- Adaptive filtering methods in the frequency domain, such as Lee or Frost filters, can effectively reduce speckle noise while preserving image details.

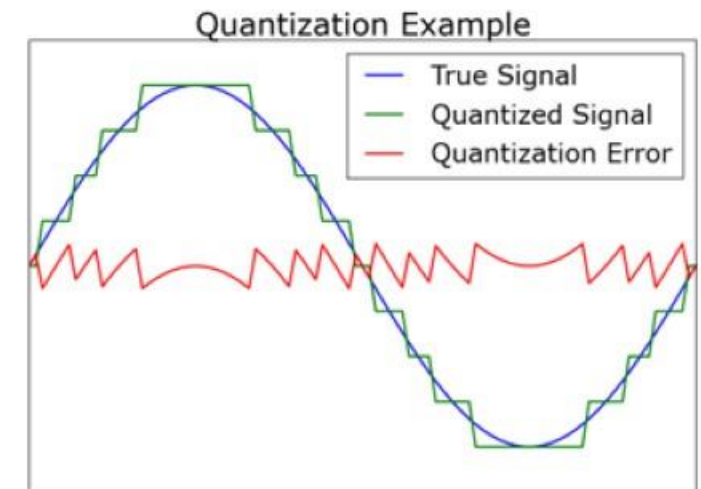


Types of Noises

In image processing, various types of noise can corrupt an image, affecting its quality and making it challenging to extract useful information.

• Quantization Noise:

- Quantization noise arises from the process of digitizing continuous signal values into discrete levels during image acquisition or processing.
- In the frequency domain, quantization noise may spread across various frequency components, depending on the quantization process.
- Applying frequency domain techniques such as dithering or error diffusion can help mitigate quantization noise and improve image quality.

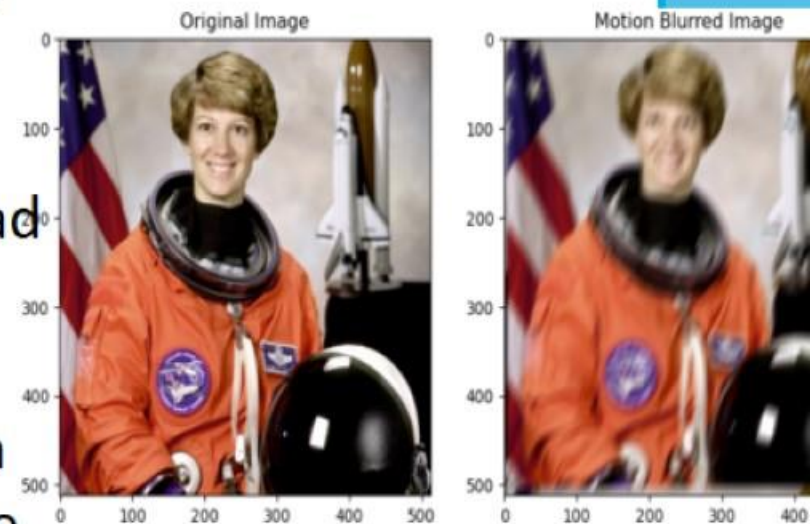


Types of Noises

In image processing, various types of noise can corrupt an image, affecting its quality and making it challenging to extract useful information.

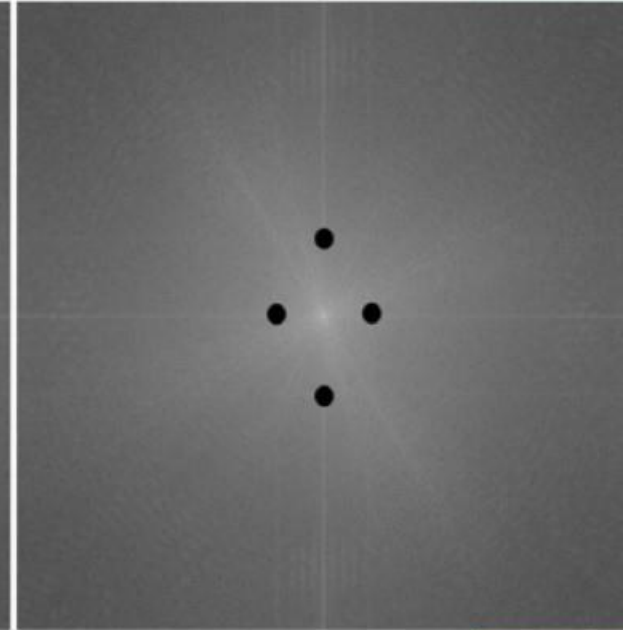
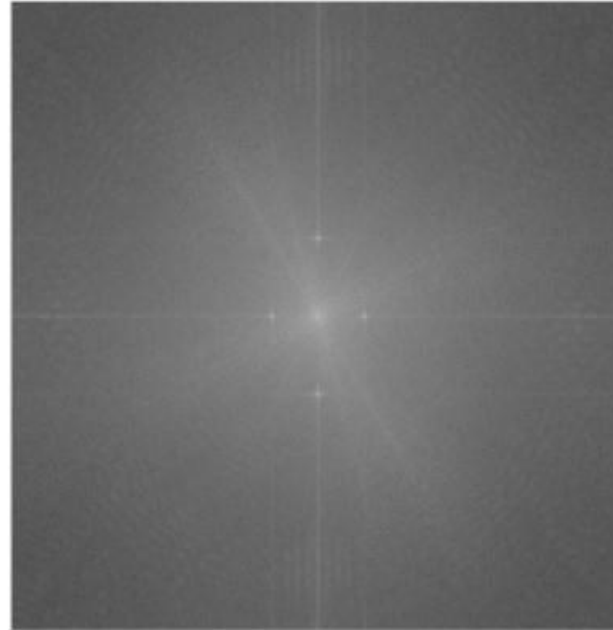
- **Blurring or Motion Blur:**

- Blurring or motion blur occurs when there is relative motion between the camera and the scene during image capture.
- In the frequency domain, blurring appears as a spread of frequency components along the direction of motion.
- Techniques like Wiener filtering or inverse filtering in the frequency domain can be used to deconvolve the blur and restore sharpness to the image.



Source: [medium](#)

Applications of Image Transformation



Source:
[craftofcoding](https://craftofcoding.com)

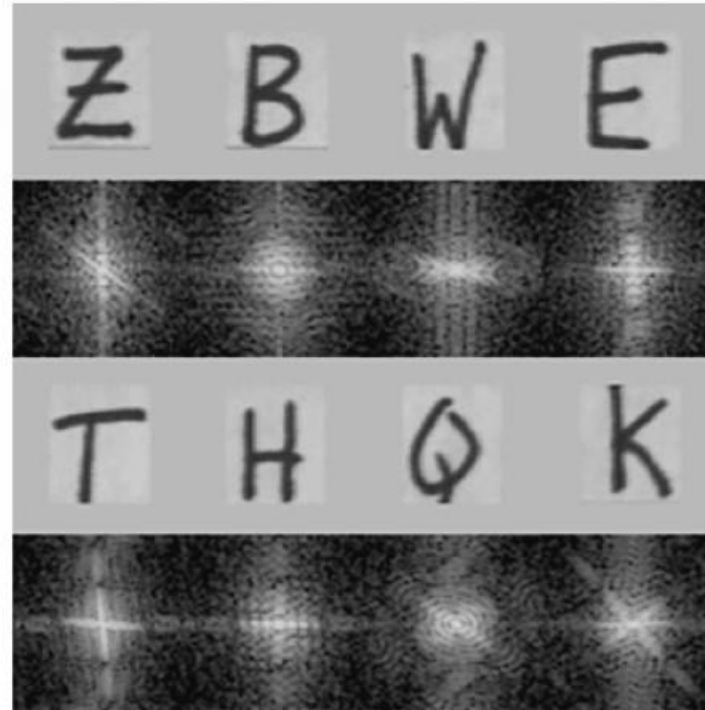
Applications of Image Transformation



Source:
[craftofcoding](https://craftofcoding.com)

Applications of Image Transformation

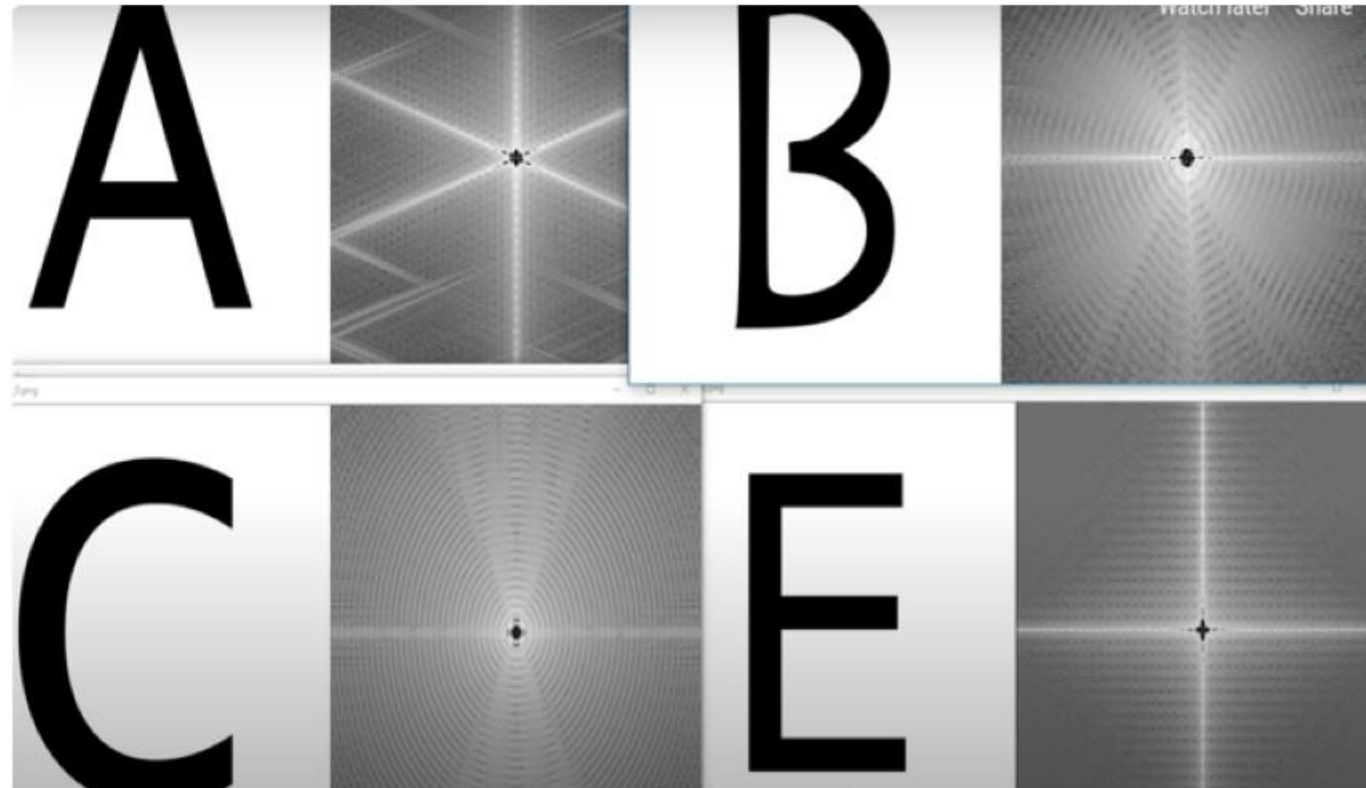
FFT in blur detection



Source:
[pyimagesearch](https://pyimagesearch.com)

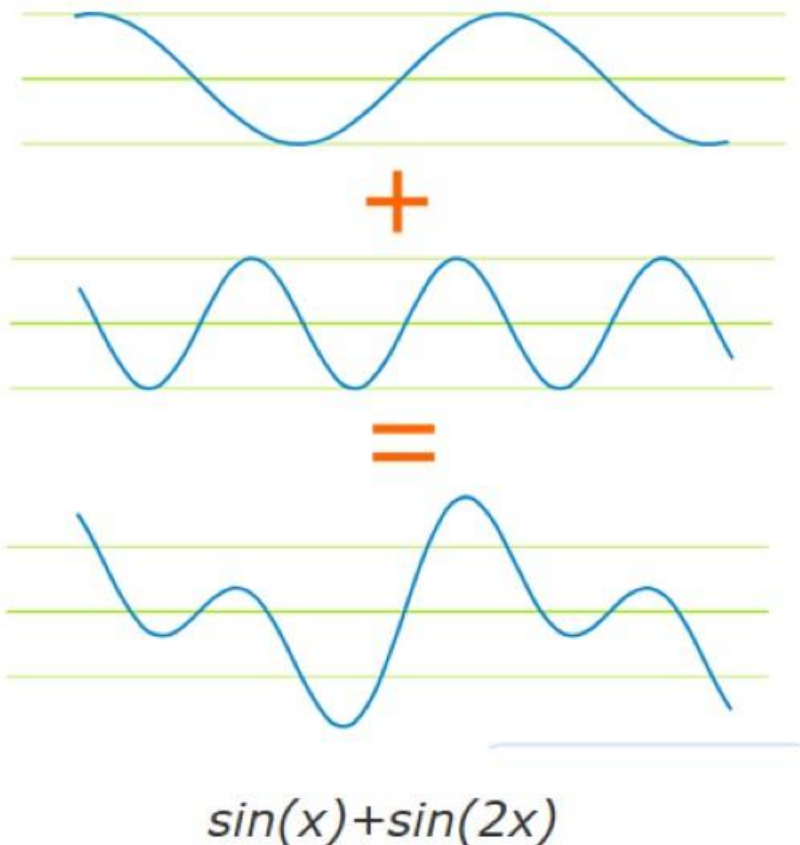
Applications of Image Transformation

FFT in Character Recognition



Source: [pysource](#)

Fourier Transform vs Fourier Series



The Fourier Series is a mathematical tool used to represent **periodic functions** as a sum of sinusoidal functions (sine and cosine) with different frequencies, amplitudes, and phase shifts. It provides a way to decompose complex periodic signals into simpler components, making it easier to analyze and manipulate them.

The Fourier transform is a mathematical technique used to analyze functions and signals in terms of their frequency components. It decomposes a function or signal into a sum of sinusoidal functions of different frequencies and amplitudes. It is used for analysis. Non-periodic signals, also known as aperiodic signals, are signals that do not repeat or oscillate in a regular pattern over time (such as the image).

2) 1-D discrete signal case

The Fourier Transform (FT) is given by: $e^{\pm j\theta} = \cos(\theta) \pm j\sin(\theta)$

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

Simple periodic patterns

▣ The **Inverse** Fourier Transform is given by:

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

Note:

➤ In the discrete case ...
the integrals are replaced by sums.

Matlab Implementation:
Functions fft, ifft, fftshift.

2) 1-D discrete signal case – example

Q: Suppose you have the following discrete signal array, convert $f(x)$ to Fourier transform?

$$f(x)$$

50	10	30	10
----	----	----	----

We use the previous formula to convert:

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

$M=4$

$$F(0) = \frac{1}{4} (f(0) * e^{-j*2*\pi*0*0/4} + f(1) * e^{-j*2*\pi*0*1/4} + f(2) * e^{-j*2*\pi*0*2/4} + f(3) * e^{-j*2*\pi*0*3/4})$$

$$= \frac{1}{4} (50 * e^0 + 10 * e^0 + 30 * e^0 + 10 * e^0)$$

$$= \frac{1}{4} (50*1 + 10*1 + 30*1 + 10*1)$$

$$= 25$$

$$F(u)$$

25			
----	--	--	--

NOTE that $F(0)$ = average of $f(x)$

2) 1-D discrete signal case – example – continued.

f(x)	50	10	30	10
------	----	----	----	----

والآن نعيد المعادلة لكل $F(u)$ لنحصل على كل القيم. حاول ان تحول $f(1)$ الى fourior حسب المعادلة

$$F(1) = \frac{1}{4} (f(0) * e^{-j*2*\pi*1*0/4} + f(1) * e^{-j*2*\pi*1*1/4} + f(2) * e^{-j*2*\pi*1*2/4} + f(3) * e^{-j*2*\pi*1*3/4})$$

$$= \frac{1}{4} (50 * e^0 + 10 * e^{-j\pi/2} + 30 * e^{-j * \pi} + 10 * e^{-j*3\pi/2})$$

Note that in the result we will have an imaginary part since there is j which means $\sqrt{-1}$

i.e. $F(u)$ consists of two parts: real $R(u)$ and imaginary $I(u)$.

The Discrete Fourier Transform (DFT)

The *Discrete Fourier Transform* of $f(x, y)$, for $x = 0, 1, 2 \dots M-1$ and $y = 0, 1, 2 \dots N-1$, denoted by $F(u, v)$, is given by the equation:

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

for $u = 0, 1, 2 \dots M-1$ and $v = 0, 1, 2 \dots N-1$.

The Steps for Fourier Transform (DFT)

- We start off by applying the Fourier Transform of



- The magnitude calculated from the complex result is shown in



The Steps for Fourier Transform (DFT)

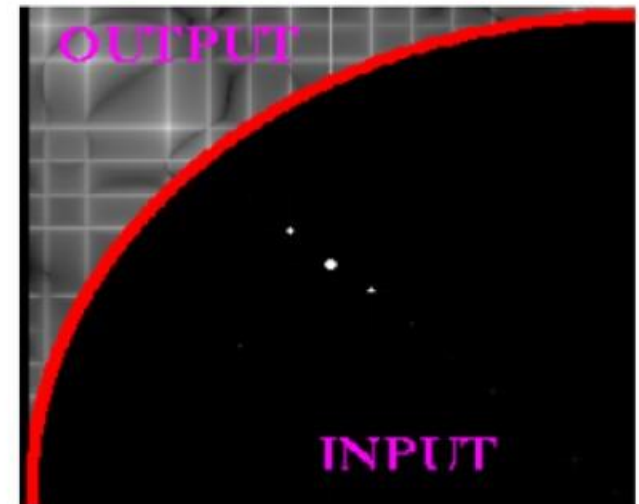
The DC-value is by far the largest component of the image.

However, the dynamic range of the Fourier coefficients (i.e. the intensity values in the Fourier image) is too large to be displayed on the screen, therefore all other values appear as black. If we apply a logarithmic transformation to the image we obtain



The Steps for Fourier Transform (DFT)

The dynamic range of an image can be compressed by replacing each pixel value with its logarithm. This has the effect that low intensity pixel values are enhanced. Applying a pixel logarithm operator to an image can be useful in applications where the dynamic range may too large to be displayed on a screen (or to be recorded on a film in the first place).



The Steps for Fourier Transform (DFT)

The result shows that the image contains components of all frequencies,

- Their magnitude gets smaller for higher frequencies. Hence, low frequencies contain more image information than the higher ones.

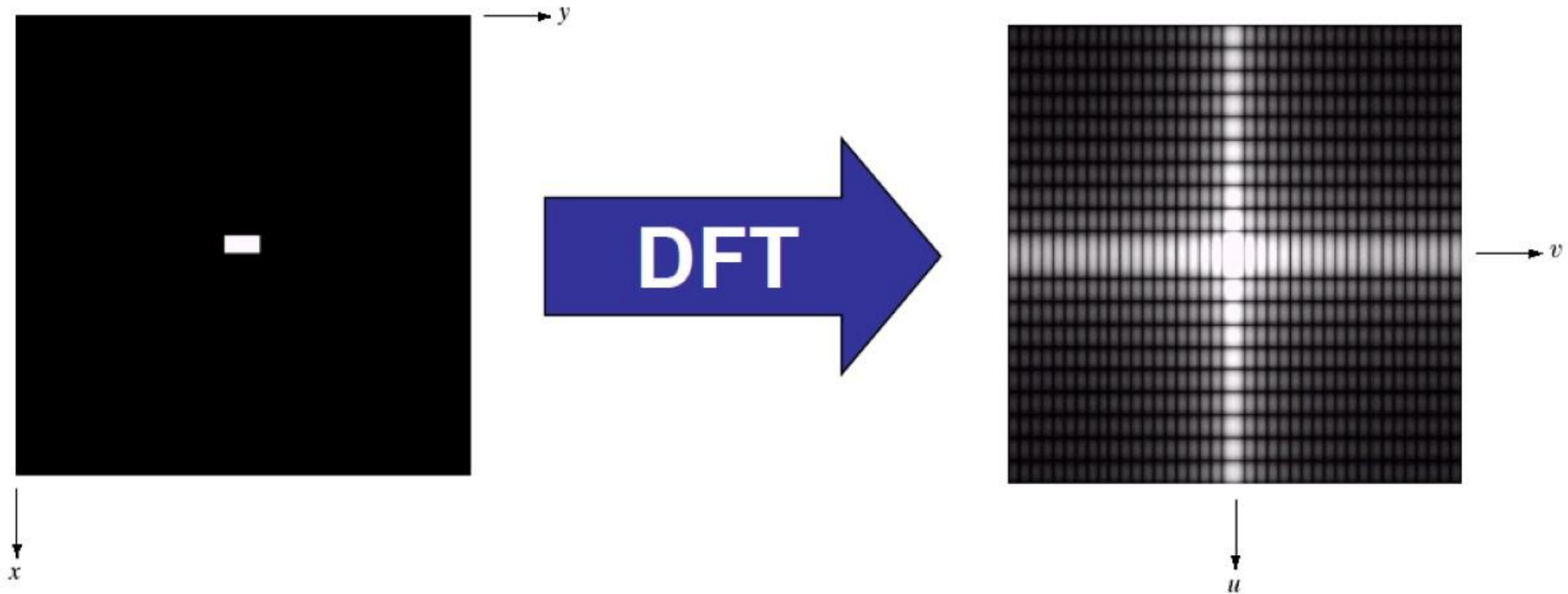
- The transform image also tells us that there are two dominating directions in the Fourier image, one passing vertically and one horizontally through the center.

- These originate from the regular patterns in the background of the original image.

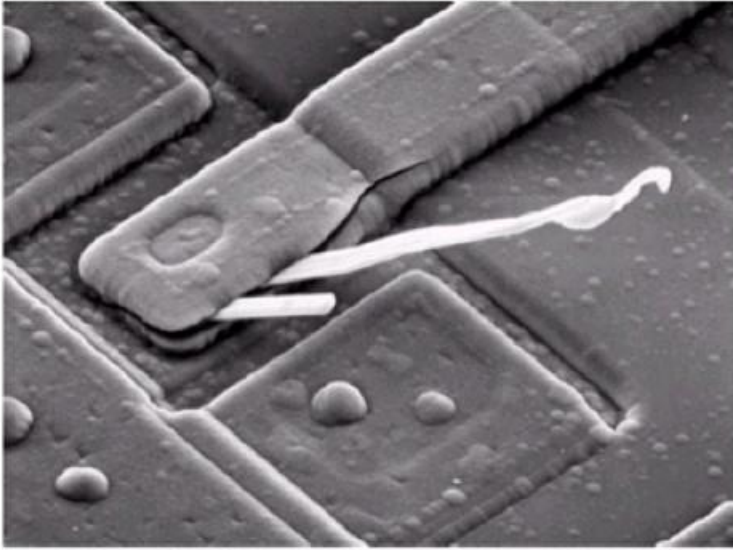


The Discrete Fourier Transform (DFT)

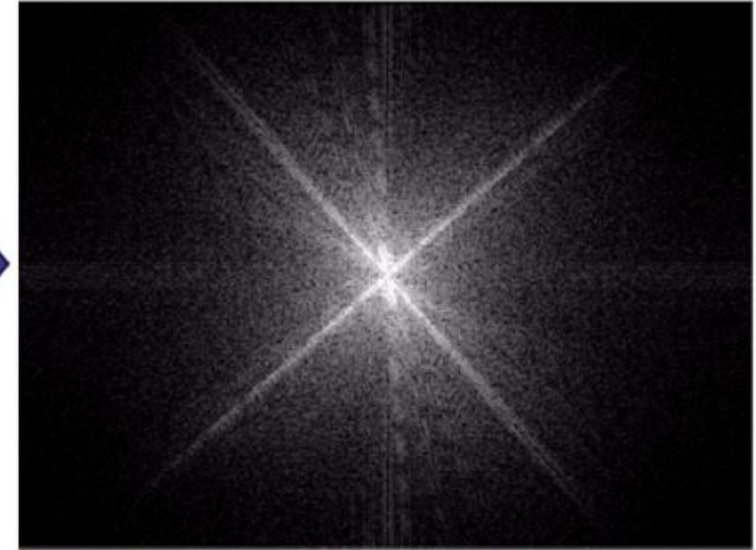
- The DFT of a two dimensional image can be visualized by showing the spectrum of the images component frequencies Images



The Discrete Fourier Transform (DFT)



Scanning electron microscope image of an integrated circuit magnified ~2500 times



Fourier spectrum of the image

Features from an image can even sometimes be seen in the Fourier spectrum of the image

The Inverse DFT

- It is really important to note that the Fourier transform is completely **reversible**
- The inverse DFT is given by:

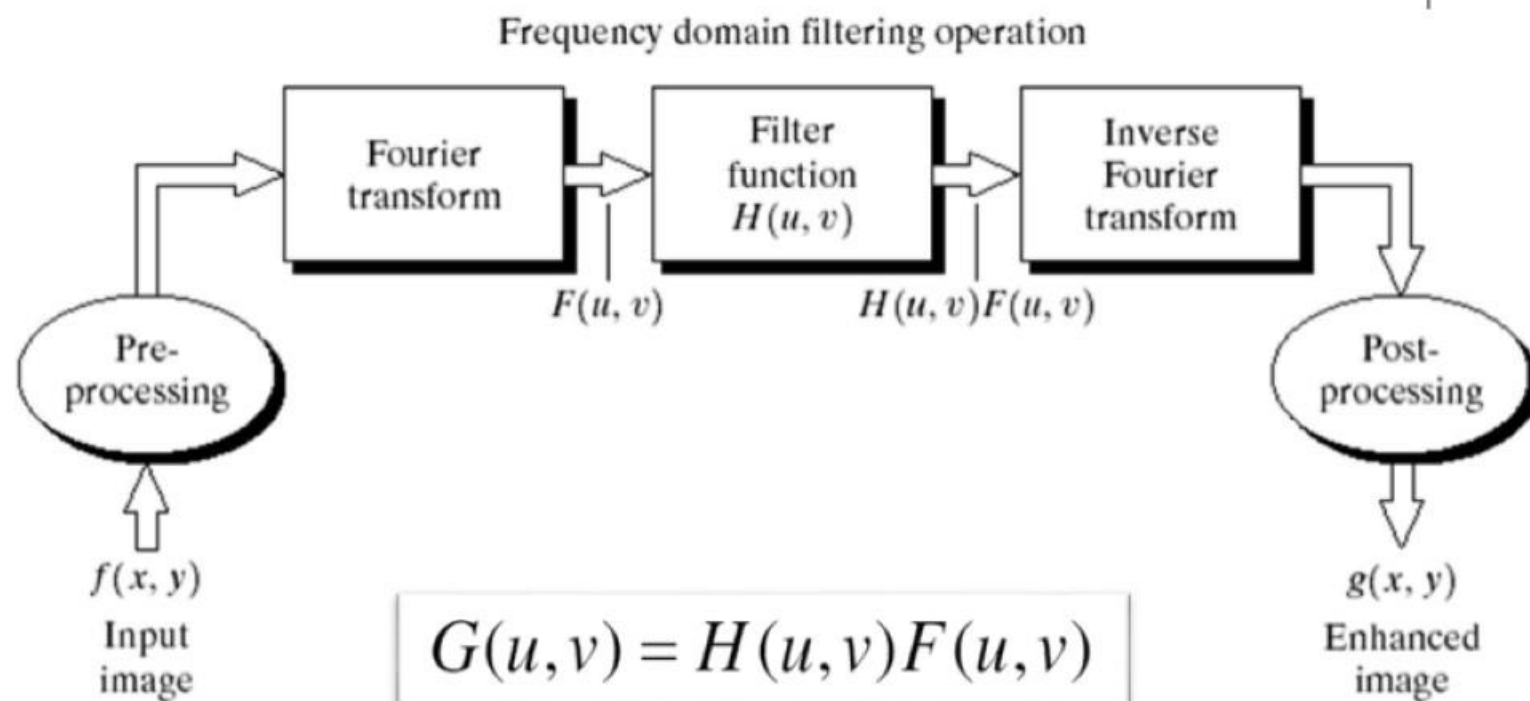
$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)}$$

for $x = 0, 1, 2 \dots M-1$ and $y = 0, 1, 2 \dots N-1$

The DFT and Image Processing

To filter an image in the frequency domain:

- 1. Compute $F(u, v)$ the DFT of the image
- 2. Multiply $F(u, v)$ by a filter function $H(u, v)$
- 3. Compute the inverse DFT of the result



Some Basic Frequency Domain Filters

$f(x,y)$

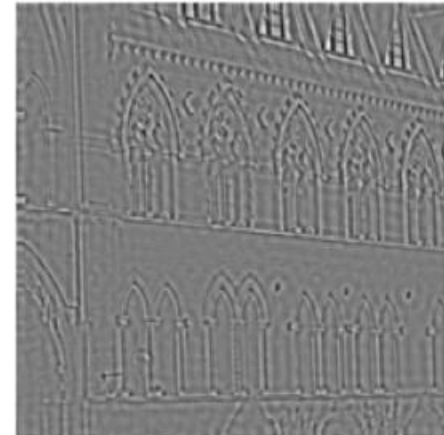
original



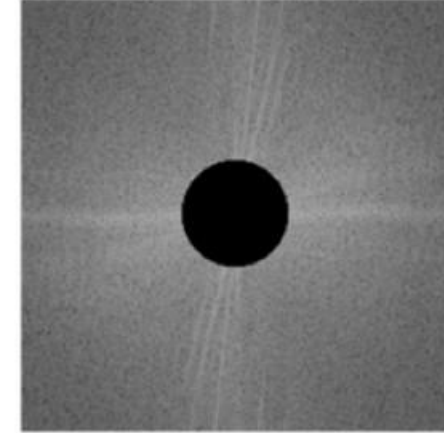
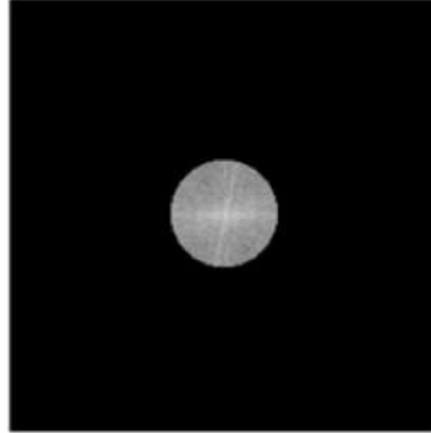
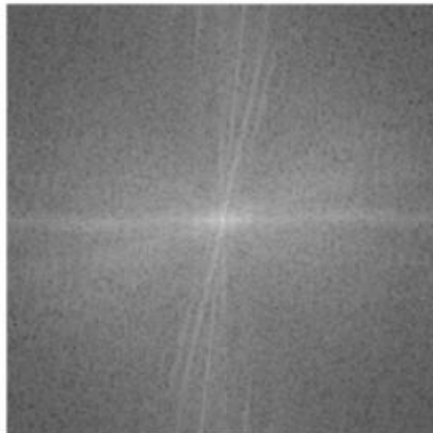
low pass



high pass



$|F(u,v)|$



Smoothing Frequency Domain Filters

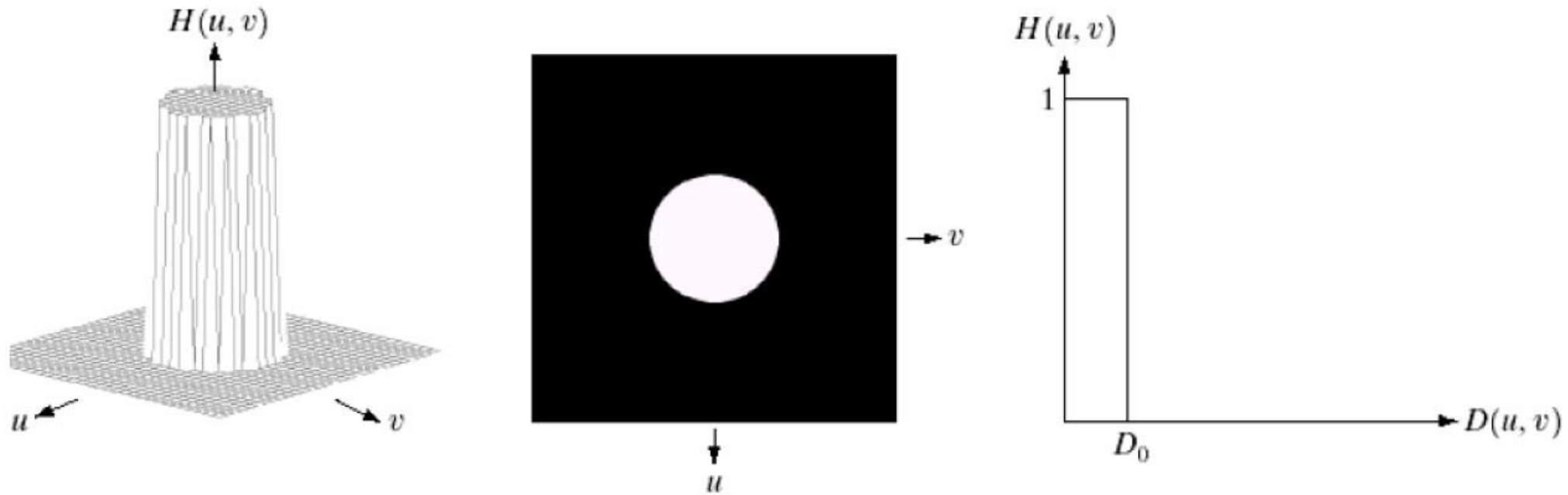
- Smoothing is achieved in the frequency domain by dropping out the high frequency components
- The basic model for filtering is:

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

- where $F(u,v)$ is the Fourier transform of the image being filtered and $H(u,v)$ is the filter transform function
- *Low pass filters* –only pass the low frequencies, drop the high ones

Ideal Low Pass Filter

- Simply cut off all high frequency components that are a specified distance D_0 from the origin of the transform



changing the distance changes the behavior of the filter

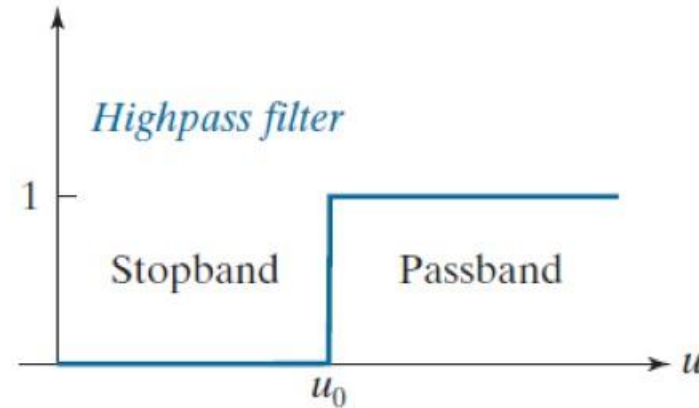
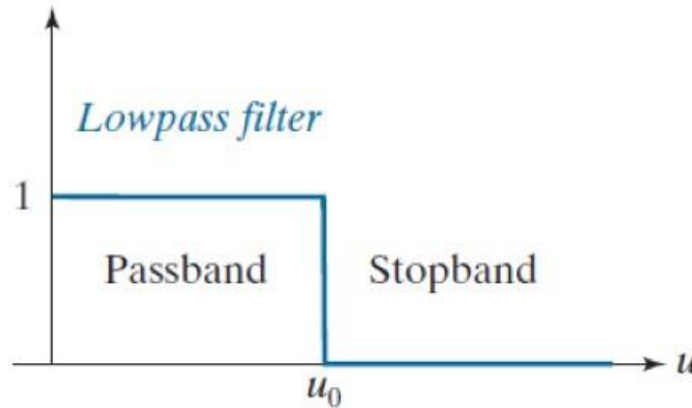
Ideal Low Pass Filter

- Simply cut off all high frequency components that are a specified distance D_0 from the origin of the transform

Transfer functions of ideal 1-D filters in the frequency domain (u denotes frequency).

(a) Lowpass filter.

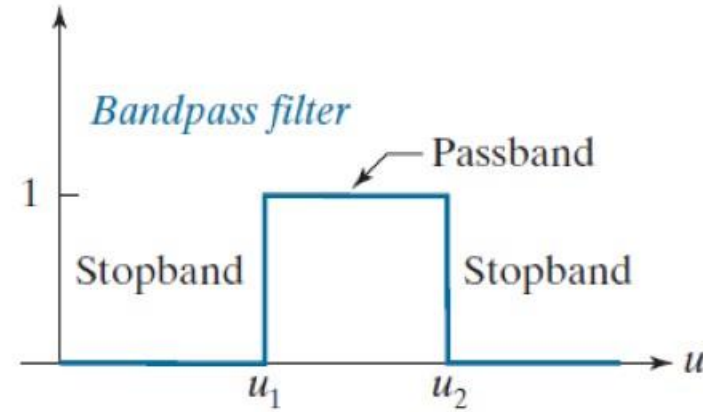
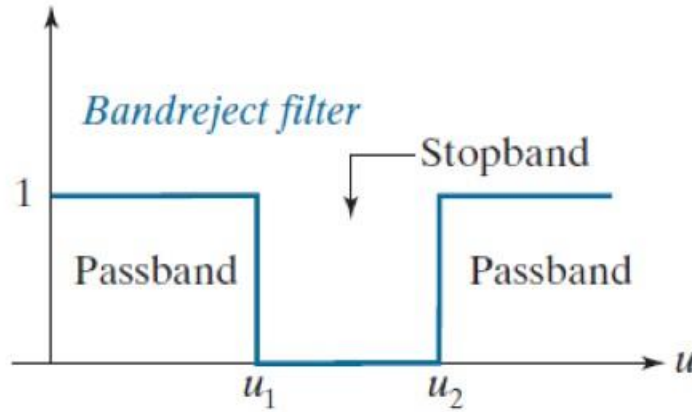
(b) Highpass filter.



Ideal Low Pass Filter

- Simply cut off all high frequency components that are a specified distance D_0 from the origin of the transform

(c) Bandreject filter.
(d) Bandpass filter.



Ideal Low Pass Filter

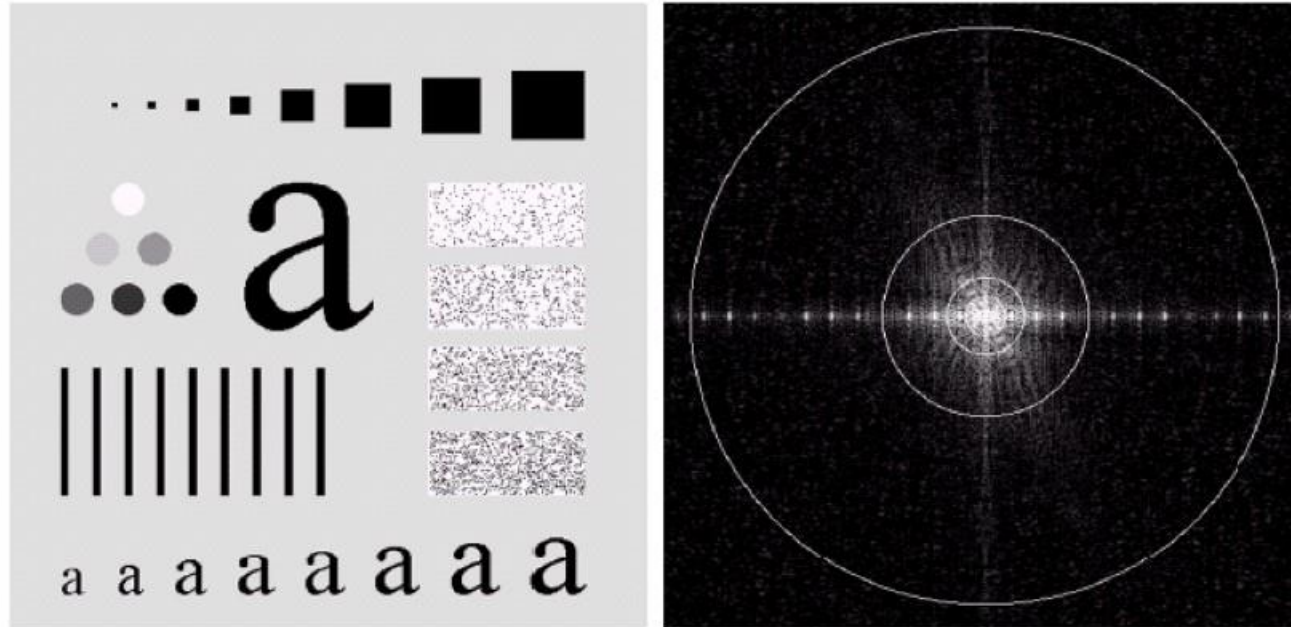
- The transfer function for the ideal low pass filter can be given as:

$$H(u, v) = \begin{cases} 1 & \text{if } D(u, v) \leq D_0 \\ 0 & \text{if } D(u, v) > D_0 \end{cases}$$

where $D(u, v)$ is given as:

$$D(u, v) = [(u - M / 2)^2 + (v - N / 2)^2]^{1/2}$$

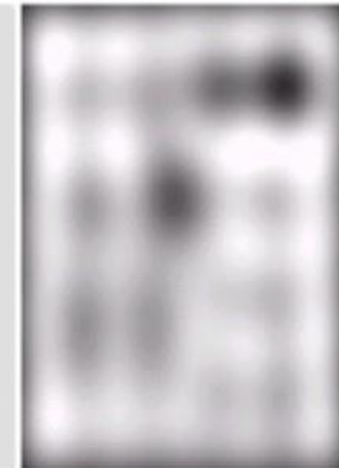
Ideal Low Pass Filter



Above we show an image, it's Fourier spectrum and a series of ideal low pass filters of radius 5, 15, 30, 80 and 230 superimposed on top of it

Ideal Low Pass Filter

Original
image



Result of filtering
with ideal low
pass filter of
radius 5

Result of filtering
with ideal low
pass filter of
radius 15



Result of filtering
with ideal low
pass filter of
radius 30

Result of filtering
with ideal low
pass filter of
radius 80

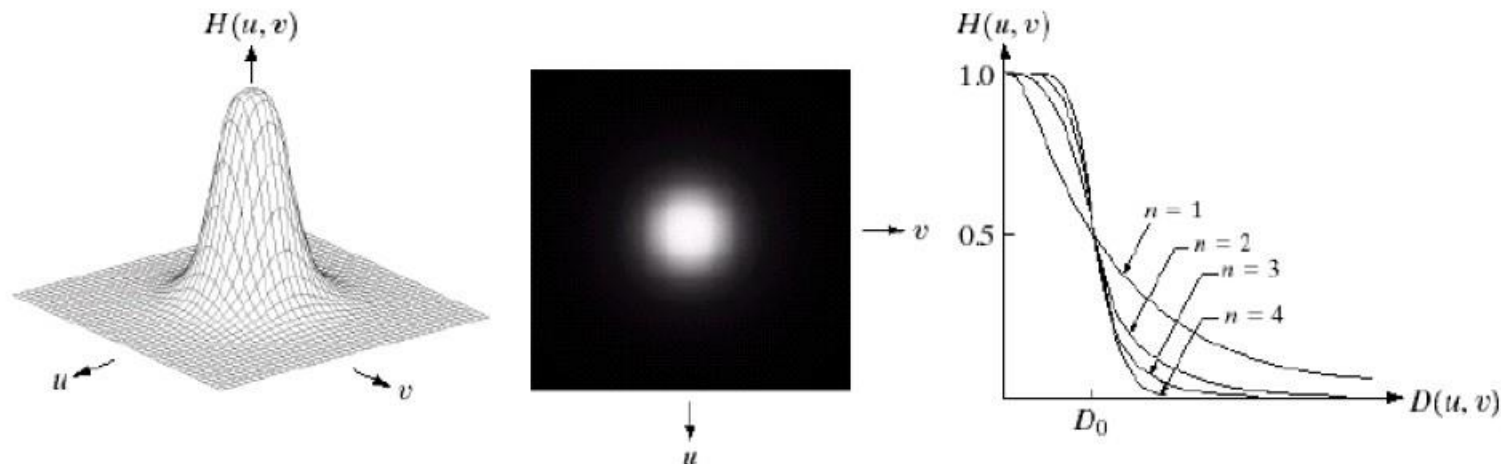


Result of filtering
with ideal low
pass filter of
radius 230

Butterworth Lowpass Filters

- The transfer function of a Butterworth lowpass filter of order n with cutoff frequency at distance D_0 from the origin is defined as:

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



Butterworth Lowpass Filters

Original
image



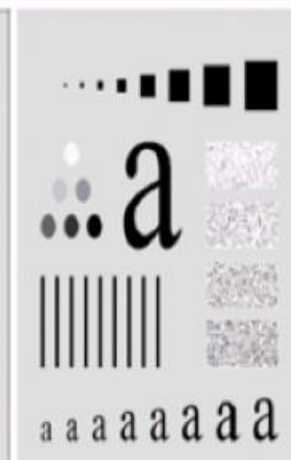
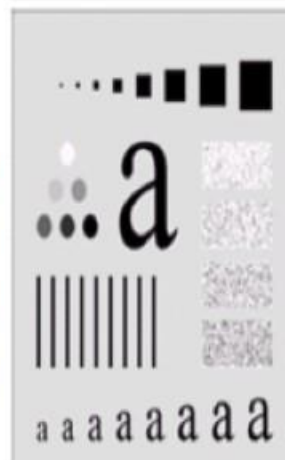
Result of filtering
with Butterworth
filter of order 2 and
cutoff radius 5

Result of filtering
with Butterworth
filter of order 2 and
cutoff radius 15



Result of filtering
with Butterworth
filter of order 2 and
cutoff radius 30

Result of filtering
with Butterworth
filter of order 2 and
cutoff radius 80

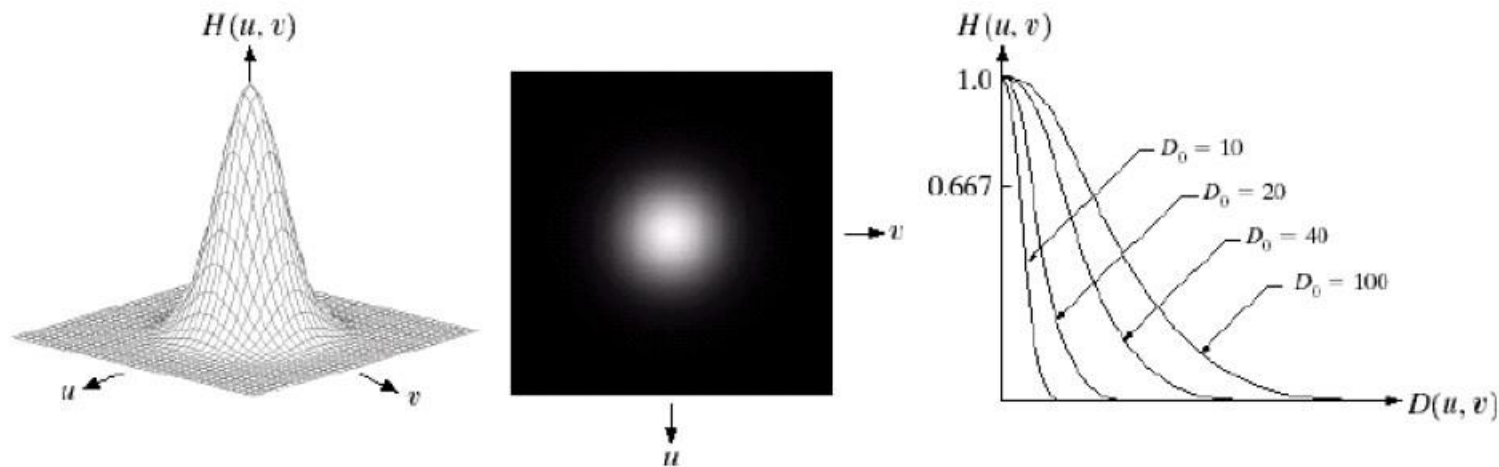


Result of filtering
with Butterworth
filter of order 2 and
cutoff radius 230

Gaussian Lowpass Filters

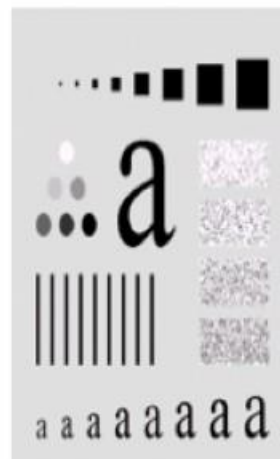
- The transfer function of a Gaussian lowpass filter is defined as:

$$H(u, v) = e^{-D^2(u, v) / 2D_0^2}$$



Gaussian Lowpass Filters

Original
image



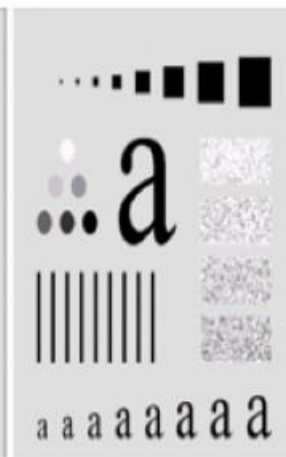
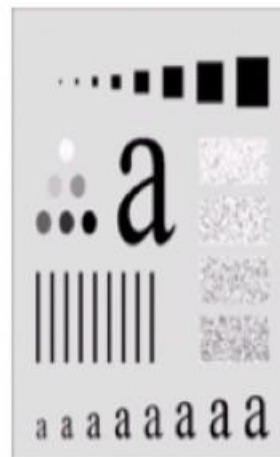
Result of filtering
with Gaussian
filter with cutoff
radius 5

Result of filtering
with Gaussian
filter with cutoff
radius 15



Result of filtering
with Gaussian
filter with cutoff
radius 30

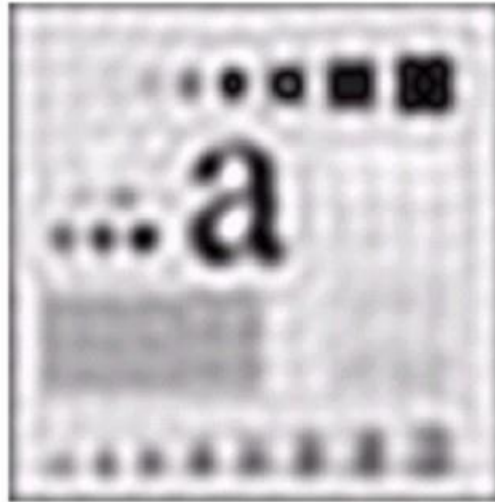
Result of
filtering with
Gaussian filter
with cutoff
radius 85



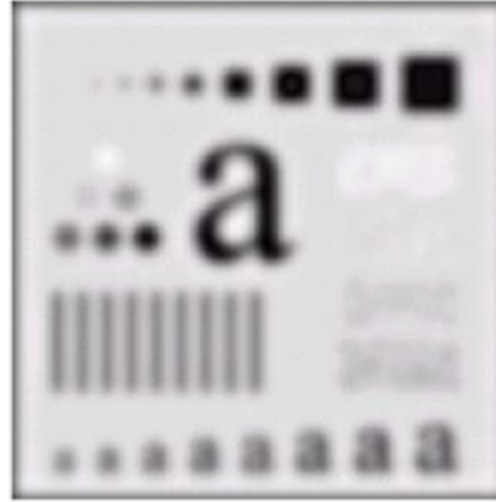
Result of filtering
with Gaussian
filter with cutoff
radius 230

Lowpass Filters Compared

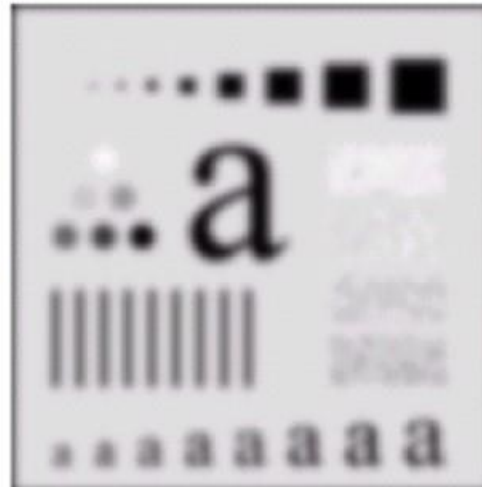
Result of filtering
with ideal low
pass filter of
radius 15



Result of
filtering with
Butterworth filter
of order 2 and
cutoff radius 15



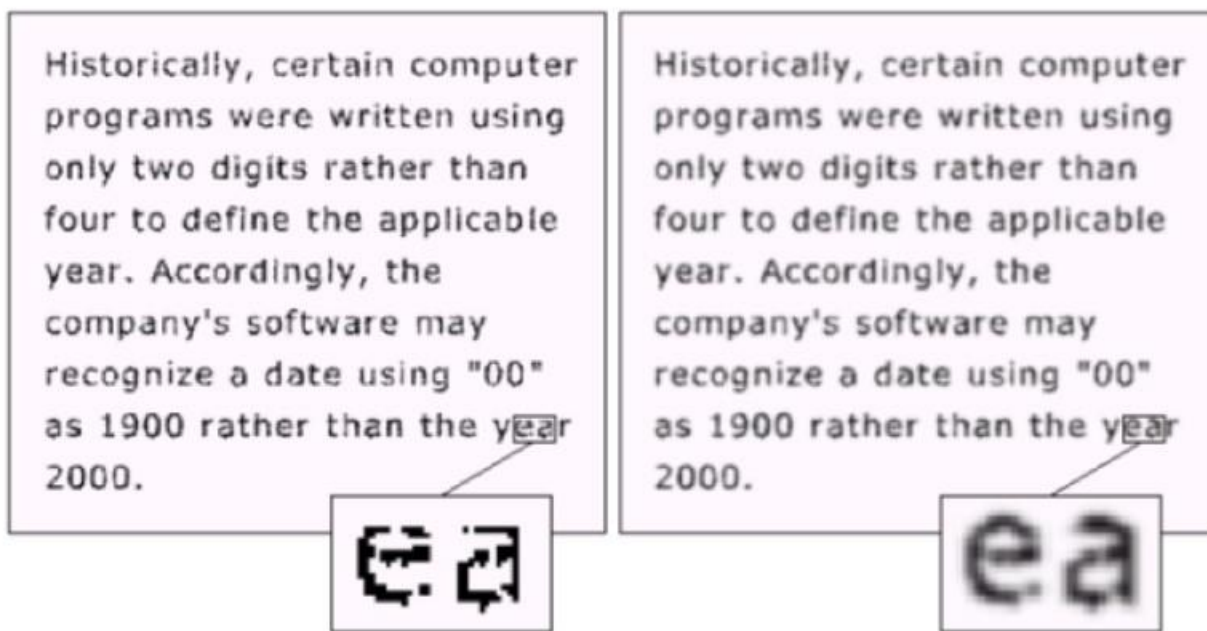
Result of filtering
with Gaussian
filter with cutoff
radius 15



Lowpass Filtering Examples

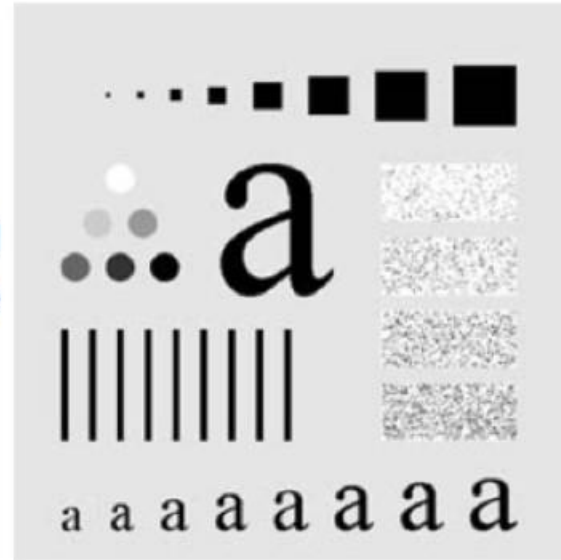
Poor Resolutions

- For the broken char. The human visual system can fill these gaps, but the M/C is not
- A low pass Gaussian filter is used to connect broken text(blurring)

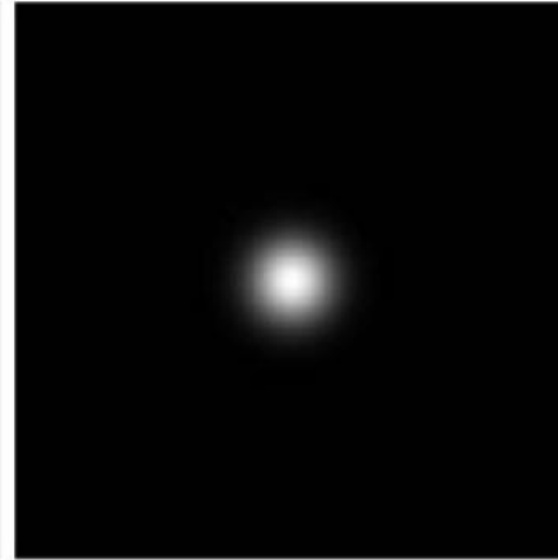


Lowpass Filtering Examples

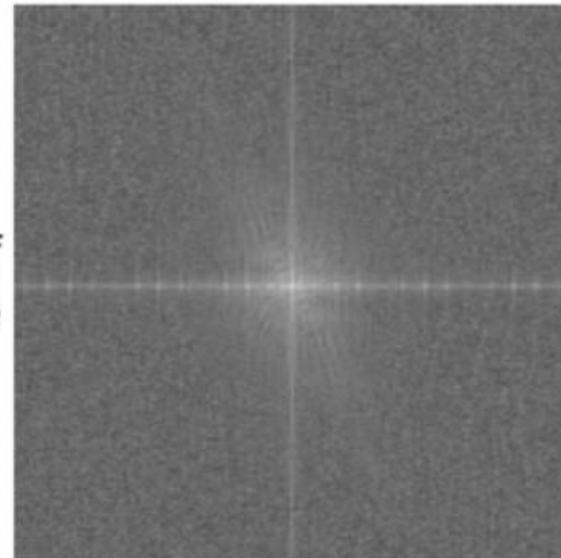
Original
image



Gaussian
lowpass filter



Spectrum of
original image



Processed
image



Lowpass Filtering Examples Cosmetic Application

Different lowpass Gaussian filters used to remove blemishes in a photograph





Thank You