Image Transform

Image Transform

- Image transform is representation of a 2D signal
  (Image signal holds 2D visual information)

- The efficient representation of visual information lies at the foundation of many image processing tasks which include image filtering, image compression and feature extraction, etc.

- Efficiency of representation: The ability to capture significant information of an image in a small description.

- Efficient image transforms are extensively used in image processing and image analysis.

- Transform is basically a mathematical tool, which allow us to move from one domain to another domain.

  (generally time/spatial domain to frequency domain)

- The reason to migrate from one domain to another domain is to perform the task at hand in an easier manner.

- Other advantages for transforming image:

1) Transformation may isolate critical components of image pattern so that they are directly accessible for analysis.

2) Transformation may place image data in a more compact form so that it can be stored and transmitted efficiently.

- Thus, image transforms are also useful for fast computation of 2D convolution and correlation.
- Transforms change the representation of the signal by projecting it on to a set of **basis functions**.
- The transforms do not change the information content present in the signal.
- The transform is reversible, i.e., we can revert to the initial domain.

- Most of the image transforms like Fourier transform, Discrete Cosine transform, Wavelet transform, etc. give information about the frequency contents in an image.

Definition:

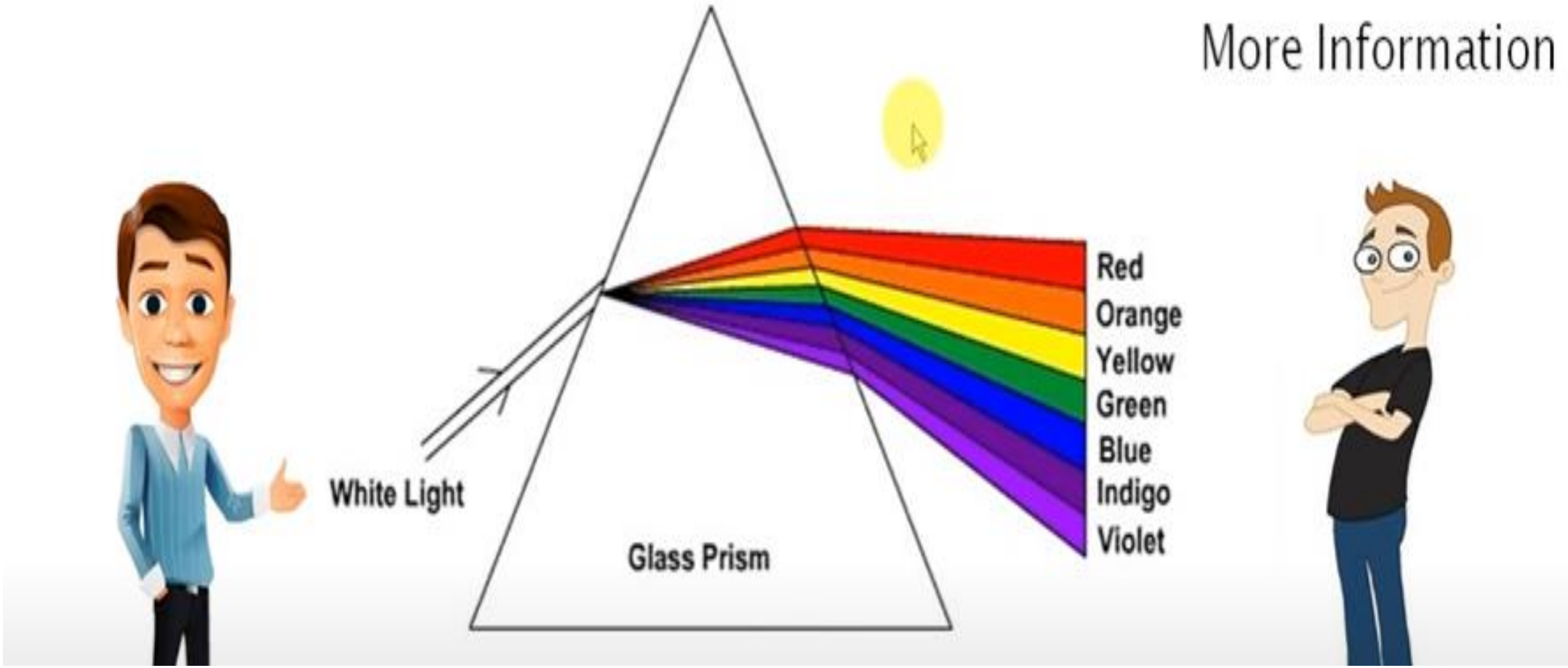
- **Image transform** = Operation to change the default representation space of a digital image (spatial domain -> another domain), so that all the information present in the image is preserved in the transformed domain, but represented differently.

- **Mathematical Convenience:**

Every action in the time domain will have an impact in the frequency domain.

e.g. The complex convolution operation in the time domain is equal to simple multiplication operation in the frequency domain.

More Information



- Based up on the nature of basis functions:

- Transforms with sinusoidal orthogonal basis functions.
- Transforms with non-sinusoidal orthogonal basis functions.
- Transforms whose basis functions depend on the statistics of the input data.
- Transforms whose basis functions are capable of representing the directional information present in the image.

- 2D Discrete Fourier Transform
- Discrete Cosine Transform
- Haar Transform
- Walsh Transform
- Hadamard Transform
- Slant Transform
- KL Transform
- Radon Transform

Fourier Transform

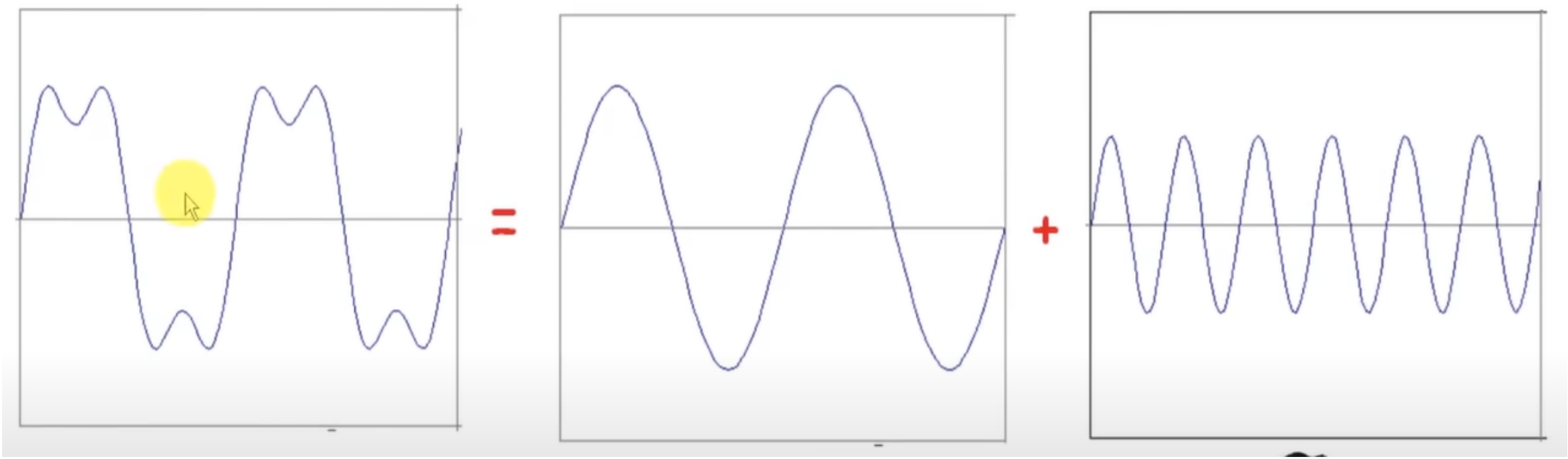
- The Fourier Transform was developed by Jean Baptiste Joseph Fourier (a French mathematician and physicist)
- It was initially developed to explain the distribution of temperature and heat conduction.
- It is widely used in image processing too.

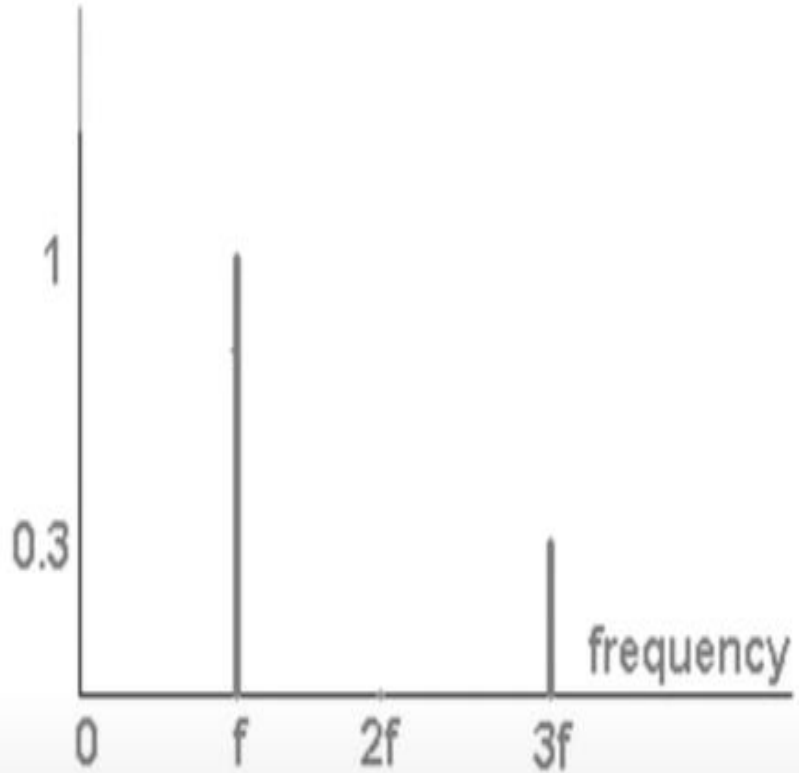
- An image is spatially varying function f(x,y).

- For analyzing spatial variations, one way is to decompose an image in to a set of orthogonal functions (Fourier functions).

- example on 1D signal:

$$g(t) = \sin(2pi\ f\ t) + (1/3)\sin(2pi\ (3f)\ t)$$





- Thus, a Fourier transform is used to transform an intensity image in to the domain of spatial frequency.

1D Fourier Transform

Represents the signal as an infinite weighted sum of an infinite number of sinusoids.

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i 2\pi u x} dx$$

Note: $e^{ik} = \cos k + i \sin k \qquad i = \sqrt{-1}$

Arbitrary function $\longrightarrow$ Single Analytic Expression
Spatial Domain (x) $\longrightarrow$ Frequency Domain (u)

Inverse Fourier Transform (IFT)

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{i 2\pi u x} dx$$

1D Fourier Transform (Cont.)

- Also, defined as:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-iux}\, dx$$

Note: $e^{ik} = \cos k + i \sin k \qquad i = \sqrt{-1}$

- Inverse Fourier Transform (IFT)

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(u) e^{iux}\, dx$$

2D Discrete Fourier Transform

- Fourier transform of a 2D signal defined over a discrete finite 2D grid of size *MxN*

or equivalently

- Fourier transform of a 2D set of samples forming a bidimensional sequence.

- As in the 1D case, 2D-DFT, though a self-consistent transform, can be considered as a mean of calculating the transform of a 2D sampled signal defined over a discrete grid.

- The signal is periodized along both dimensions and the 2D-DFT can be regarded as a sampled version of the 2D DTFT.

2D DFT (Cont.)

- 2D Fourier (discrete time) Transform (DTFT)

$$F(u,v) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n] e^{-j2\pi(um+vn)}$$

- 2D Discrete Fourier Transform (DFT)

$$F[k,l] = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-j2\pi\left(\frac{k}{M}m+\frac{l}{N}n\right)}$$

Properties of 2-D DFT

1. Separable Property

- It allows a 2D transform to be computed in two steps by successive 1D operations on rows and columns of image.

- The main advantage of separability property is that a Fourier transform of any dimensions can be performed by applying a 1D transform on each dimensions.

Properties of 2-D DFT

1. Separable Property

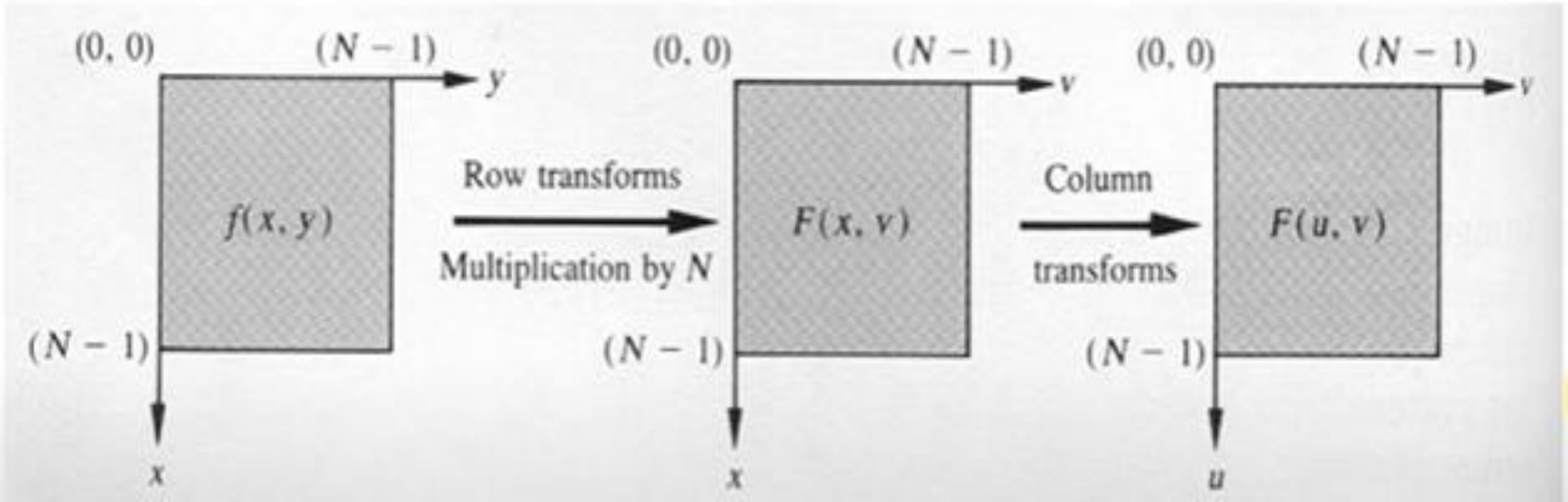
$$F(u,v) = \frac{1}{M}\sum_{x=0}^{M-1} F(x,v)\exp[-j2\pi ux/M]$$

Where,

$$F(x,v) = \left[\frac{1}{N}\sum_{y=0}^{N-1} f(x,y)\exp[-j2\pi vy/N]\right]$$

Properties of 2-D DFT

1. Separable Property



Properties of 2-D DFT (Cont.)

2. Spatial Shift Property

$$f(x,y)\exp[j2\pi(u_0x/M+v_0y/N)]\Leftrightarrow F(u-u_0,v-v_0)$$

and

$$f(x-x_0,y-y_0)\Leftrightarrow F(u,v)\exp[-j2\pi(ux_0/M+vy_0/N)]$$

This proves that the DFT of a shifted function is unaltered except for a linearly varying phase function.

Properties of 2-D DFT (Cont.)

3. Periodicity Property

- A [M,N] point DFT is periodic with period [M,N]

$$F[u,v] = F[u+mM,v] = F[u,v+nN] = F[u+mM,v+nN]$$

$$f[k,l] = f[k+mM,l] = f[k,l+nN] = f[k+mM,l+nN]$$

- This has important consequences on the implementation and energy compaction property.

Properties of 2-D DFT (Cont.)

4. Convolution Property

- This property tells that the convolution of two functions in the spatial domain corresponds to multiplication in the frequency domain and vice versa.

$$f(x) * g(x) \Leftrightarrow F(u)G(u)$$

$$f(x)g(x) \Leftrightarrow F(u) * G(u)$$

Properties of 2-D DFT (Cont.)

5. Correlation Property

- Correlation is basically used to find the relative similarity between two signals.
- The process of finding similarity of a signal to itself is autocorrelation.
- The process of finding similarity of two different signals is cross correlation.

Properties of 2-D DFT (Cont.)

5. Correlation Property

- This property tells that the correlation of two sequences in time domain is equal to the multiplication of DFT of one sequence and time reversal of the DFT of another sequence in the frequency domain.

$$f(x,y) \circ g(x,y) \Leftrightarrow F^*(u,v)G(u,v)$$

$$f^*(x,y)g(x,y) \Leftrightarrow F(u,v) \circ G(u,v)$$

Properties of 2-D DFT (Cont.)

6. Scaling Property

$$af(x,y) \Leftrightarrow aF(u,v)$$

$$f(ax,by) \Leftrightarrow \frac{1}{|ab|} F(u/a, v/b)$$

- According to this property, the expansion of signal in one domain is equal to compression of signal in another domain.

7. Conjugate Symmetry Property

$$DFT[f^*(m,n)] = F^*(-k,-l)$$

$$F(k,l) = F^*(-k,-l)$$

8. Orthogonality Property

- The orthogonality condition can be used to derive the formula for the IDFT from the definition of DFT.

9. Multiplication by Exponential

- This property tells that multiplication of a function f(m,n) with an exponential in the spatial domain leads to a frequency shift.

10. Rotation Property

- Polar coordinates:

$$x = r\cos\theta, \quad y = r\sin\theta, \quad u = \omega\cos\varphi, \quad v = \omega\cos\varphi$$

Which means that:

$$f(x,y), F(u,v) \text{ become } f(r,\theta), F(\omega,\varphi$$

$$f(r,\theta+\theta_0) \Leftrightarrow F(\omega,\varphi+\theta_0)$$

- This property states that if a function is rotated by the angle, its Fourier transform also rotates by an equal amount (and vice versa).

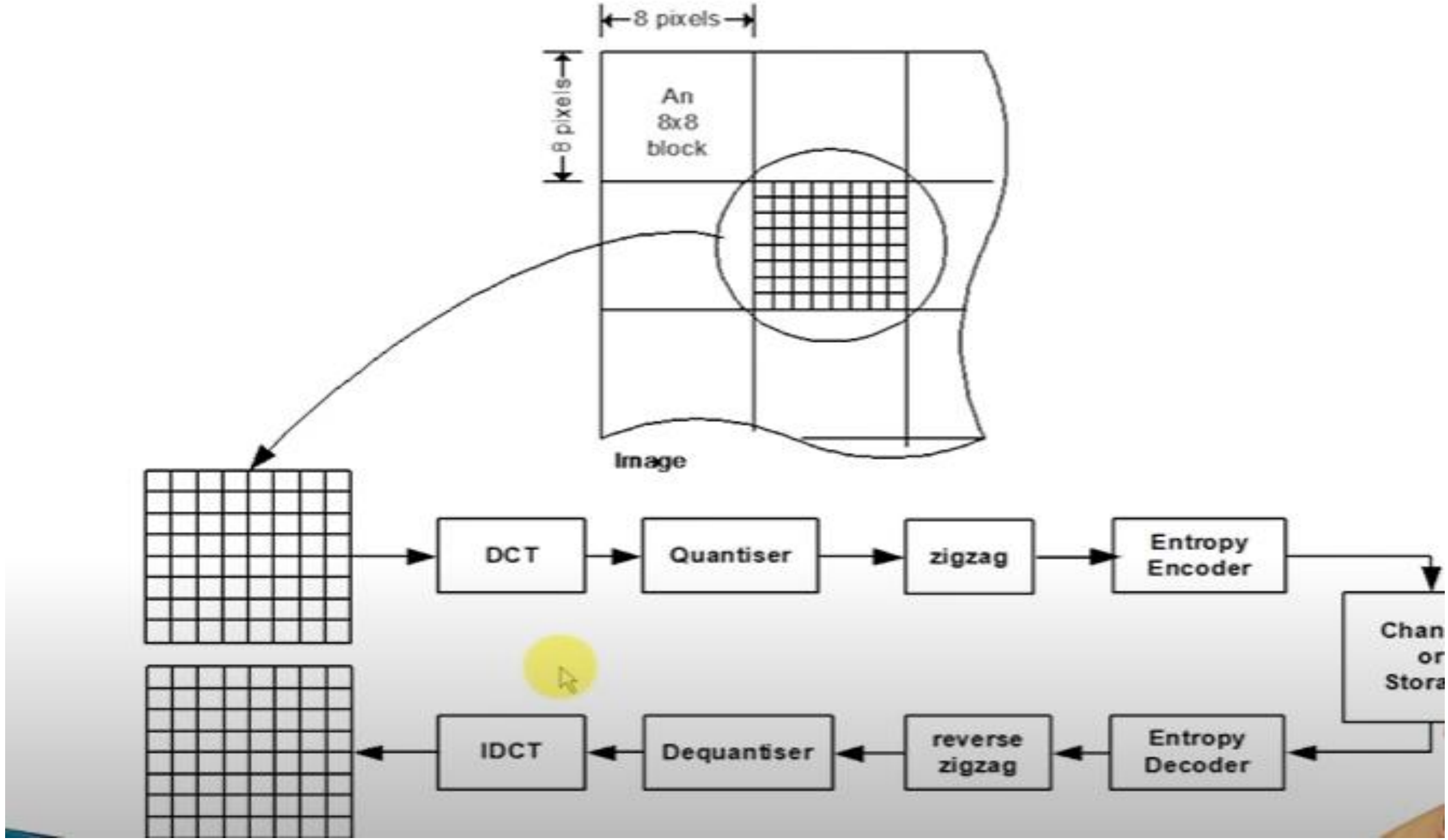
Discrete Cosine Transform

- DFT is very popular due to its good computational efficiency.
- However, it has strong disadvantages for some applications:

1) it is complex
2) it has comparable poor energy compaction, etc

- The Discrete Cosine Transform was developed by Ahmed, Natarajan and Rao in 1974.
- The discrete cosine transform is the member of the family of real valued discrete sinusoidal unitary transforms.
- The discrete cosine transform consists of a set of basis vectors that are sampled cosine functions.
- DCT is a technique for converting a signal in to elementary frequency components and is widely used in image compression.

- The ability to pack the energy of the spatial sequence into as few frequency coefficients as possible is called as Energy compaction.
- This is very important for image compression.
- Here, we represent the signal in the frequency domain.
- If compaction is high, we only have to transmit a few coefficients.
- Instead of the whole set of pixels.


8 pixels
8 pixels
An
8x8
block
Image
DCT
Quantiser
zigzag
Entropy
Encoder
Chan
or
Stora
IDCT
Dequantiser
reverse
zigzag
Entropy
Decoder

- The coefficients are then reordered into a one-dimensional array in a zigzag manner before further entropy encoding.

- The compression is achieved in two stages; the first is during quantization and the second during the entropy coding process.

- JPEG decoding is the reverse process of coding.

Discrete Cosine Transform

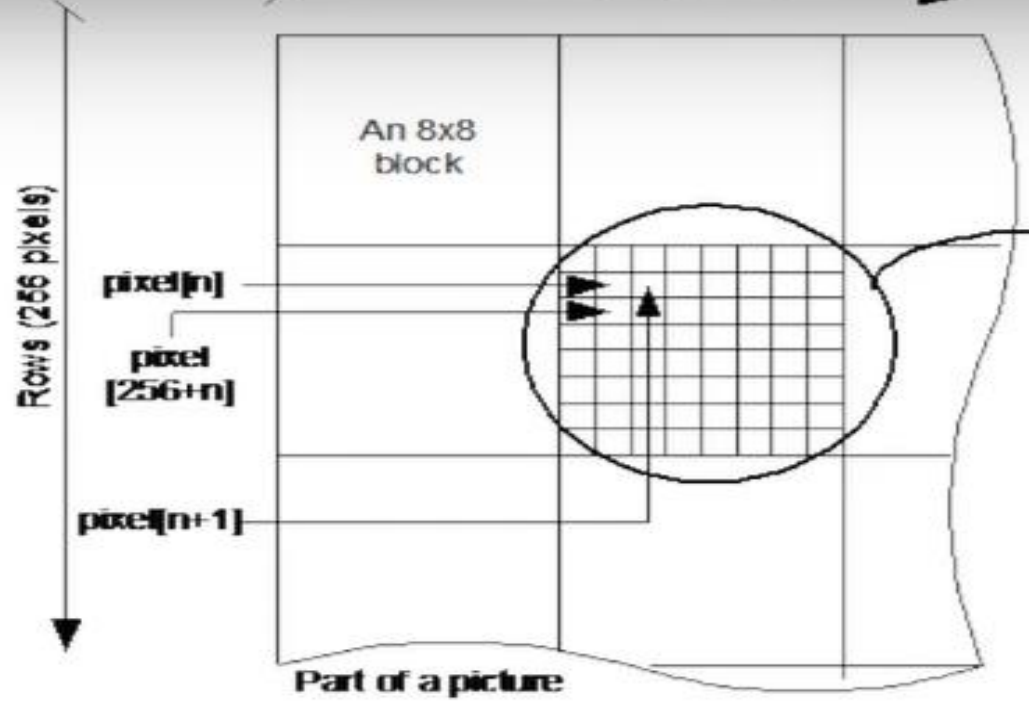
- DCT-based codecs use a two-dimensional version of the transform.
- The 2-D DCT and its inverse (IDCT) of an N x N block are shown below:
- 2-D DCT: $$F(u,v)=\frac{2}{N}C(u)C(v)\sum_{y=0}^{N-1}\sum_{x=0}^{N-1}f(x,y)\cos[\frac{(2x+1)u\pi}{2N}]\cos[\frac{(2y+1)v\pi}{2N}]$$
- 2-D IDCT:

$$f(x,y)=\frac{2}{N}\sum_{v=0}^{N-1}\sum_{u=0}^{N-1}C(u)C(v)F(u,v)\cos[\frac{(2x+1)u\pi}{2N}]\cos[\frac{(2y+1)v\pi}{2N}]$$

- The DCT is similar to the DFT since it decomposes a signal into a series of harmonic cosine functions.

Discrete Cosine Transform

- One of the properties of the 2-D DCT is that it is separable meaning that it can be separated into a pair of 1-D DCTs.
- To obtain the 2-D DCT of a block a 1-D DCT is first performed on the rows of the block then a 1-D DCT is performed on the columns of the resulting block.
- The same applies to the IDCT.
- This process is illustrated as:



Raw Block of Pixels

128	127	130	128	134	130	128	128
128	127	130	128	134	130	128	128
125	126	130	127	130	131	128	127
124	128	127	126	129	130	127	127
131	132	130	129	132	128	129	131
129	131	134	131	133	129	131	131
129	134	134	136	136	137	134	132
133	135	136	138	137	140	135	132

Result after DCT by rows

365	-1	-4	1	1	-3	1	3
365	-1	-4	1	1	-3	1	3
362	-3	-4	0	-3	-1	2	1
359	-3	-3	0	-3	-3	0	-1
368	1	1	-1	1	-3	-2	1
370	0	-3	-3	-1	-1	-1	2
378	-3	-6	0	-3	-2	-1	-2
383	-1	-6	2	-3	0	0	-3

Final result after DCT by columns

1024	-4	-11	0	-4	-6	0	1
-18	-1	1	0	2	-2	2	4
11	-1	-5	2	0	0	1	-1
0	2	2	-3	3	-2	-2	3
0	1	1	1	0	-1	-1	-2
-4	-3	-3	-1	-3	-1	0	-2
-2	0	-1	-2	-1	1	1	0
2	0	1	0	0	-1	-1	-1

2-D DCT using a 1-D DCT Pair

- 1-D DCT: $$X(k) = \sqrt{\frac{2}{N}} C(k) \sum_{i=0}^{N-1} x(i) \cos\left[\frac{(2i+1)k\pi}{2N}\right]$$

- 1-D IDCT: $$x(i) = \sqrt{\frac{2}{N}} \sum_{k=0}^{N-1} C(k) X(k) \cos\left[\frac{(2i+1)k\pi}{2N}\right]$$

k = 0, 1, 2, ..., N-1.
and i = 0, 1, 2, ..., N-1.

Limitations of DCT

- Despite excellent energy compaction capabilities, DCT offers a few limitations which restrict its use in very low bit rate applications.

- The limitations are listed below:

(i) Truncation of higher spectral coefficients results in blurring of the images, especially wherever the details are high.

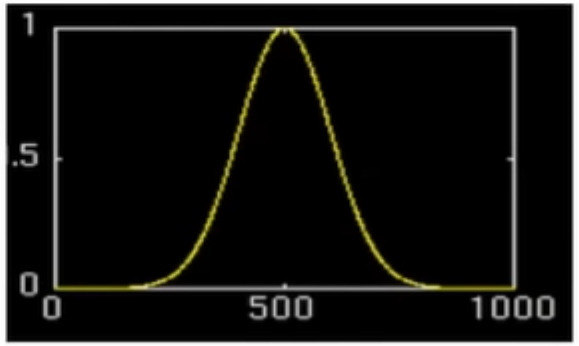
(ii) Coarse quantization of some of the low spectral coefficients introduces graininess in the smooth portions of the images.

(iii) Serious blocking artifacts are introduced at the block boundaries, since each block is independently encoded, often with a different encoding strategy and the extent of quantization.

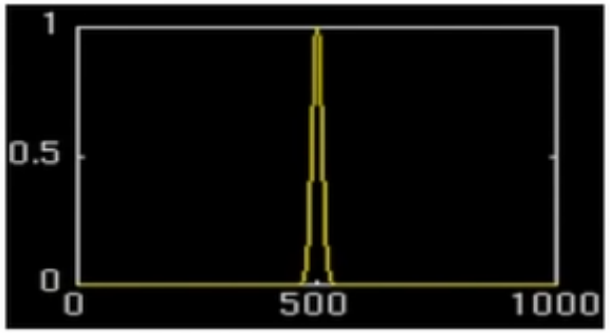
Topic Name : Haar Transform

STFT

- Time – Frequency localization depends on window size.
  - Wide window → good frequency localization, poor time localization.



  - Narrow window → good time localization, poor frequency localization.

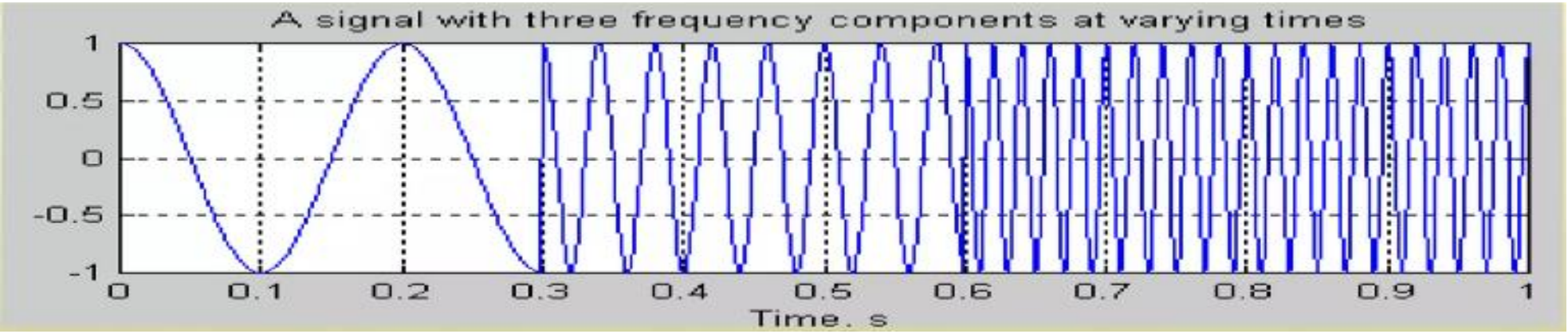


Wavelet Transform

- Uses a variable length window

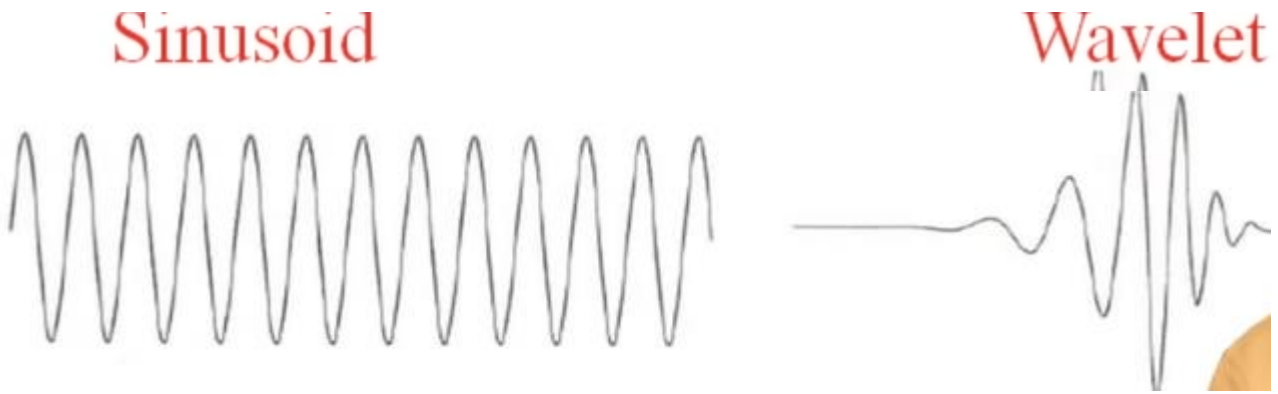
e.g.:

- Narrower windows are more appropriate at high frequencies
- Wider windows are more appropriate at low frequencies



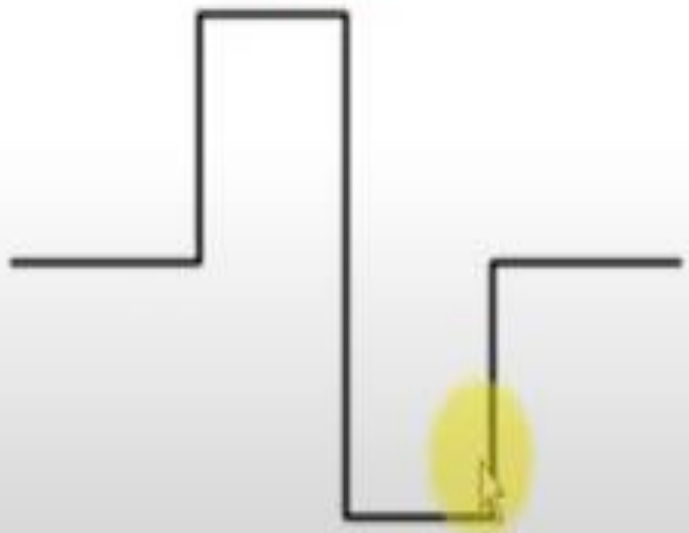
What is a wavelet?

- A function that "waves" above and below the x-axis with the following properties:
  - Varying frequency
  - Limited duration
  - Zero average value
- This is in contrast to sinusoids, used by FT, which have infinite duration and constant frequency.



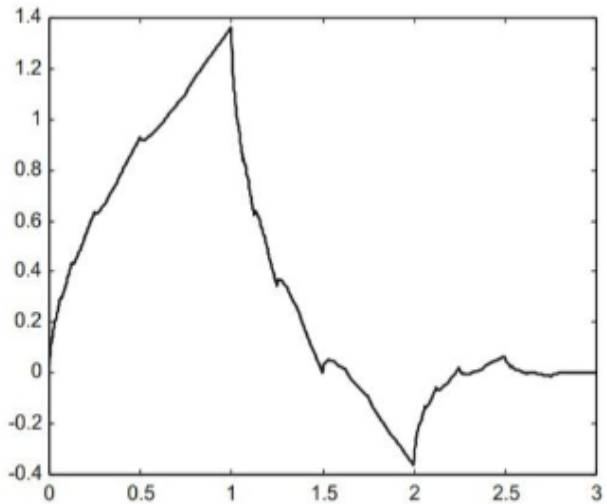
- There are many different wavelets, for example:

Haar



Morlet

Daubechies



Continuous Wavelet Transform

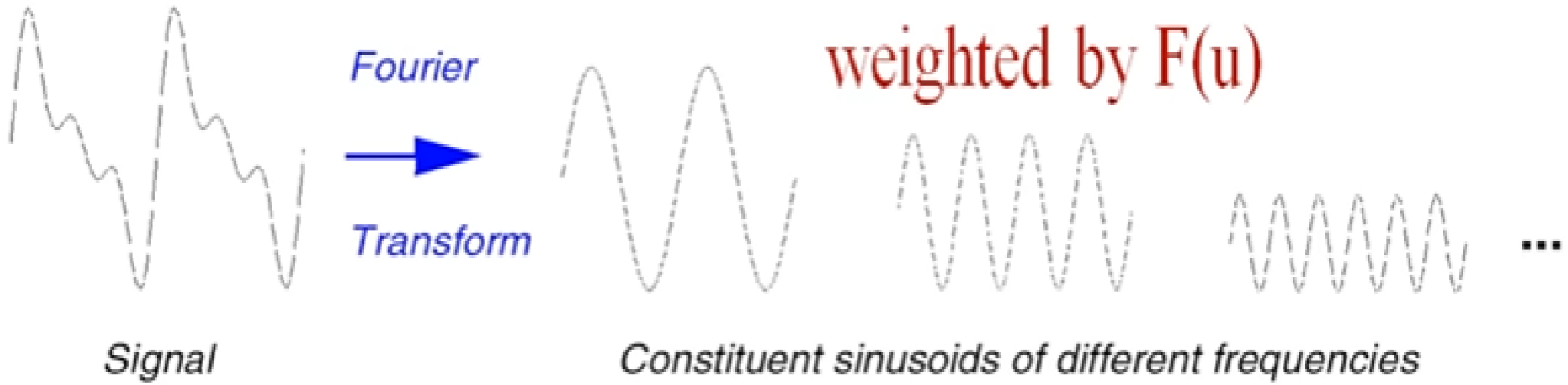
Forward CWT:

$$C(\tau, s) = \frac{1}{\sqrt{s}} \int_t f(t)\psi^*\left(\frac{t-\tau}{s}\right)dt$$

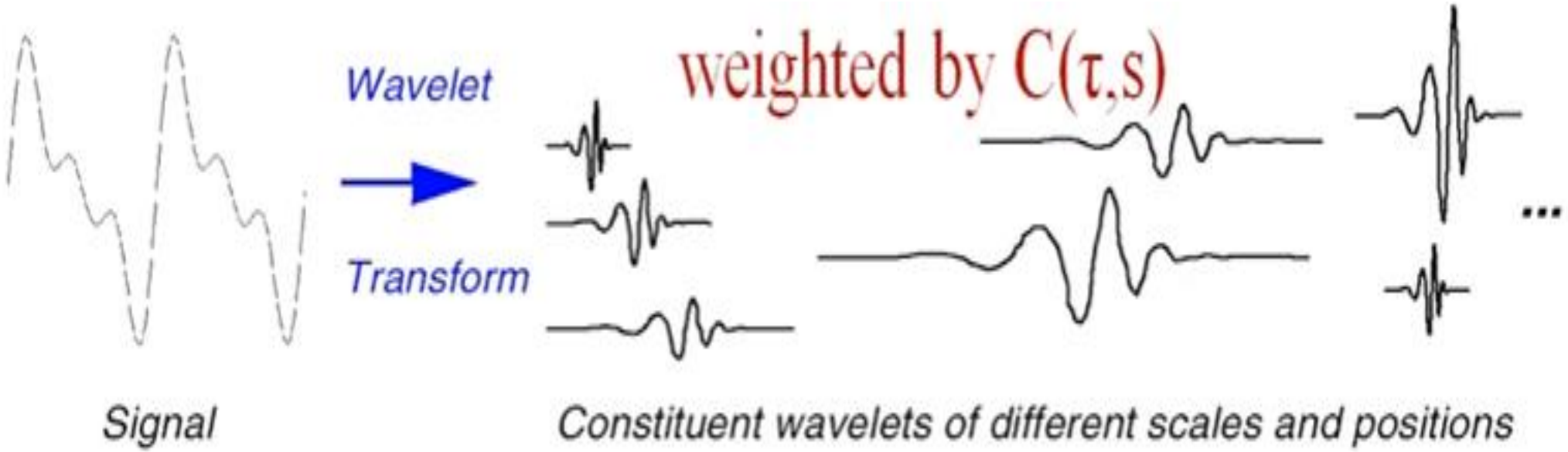
Inverse CWT:

$$f(t) = \frac{1}{\sqrt{s}} \int_\tau \int_s C(\tau, s)\psi\left(\frac{t-\tau}{s}\right)d\tau ds$$

Fourier Transform vs Wavelet Transform



$$f(x) = \int_{-\infty}^{\infty} F(u) e^{j2\pi ux} du$$



$$f(t) = \frac{1}{\sqrt{s}} \int_{\tau} \int_{s} C(\tau, s) \psi\left(\frac{t-\tau}{s}\right) d\tau ds$$

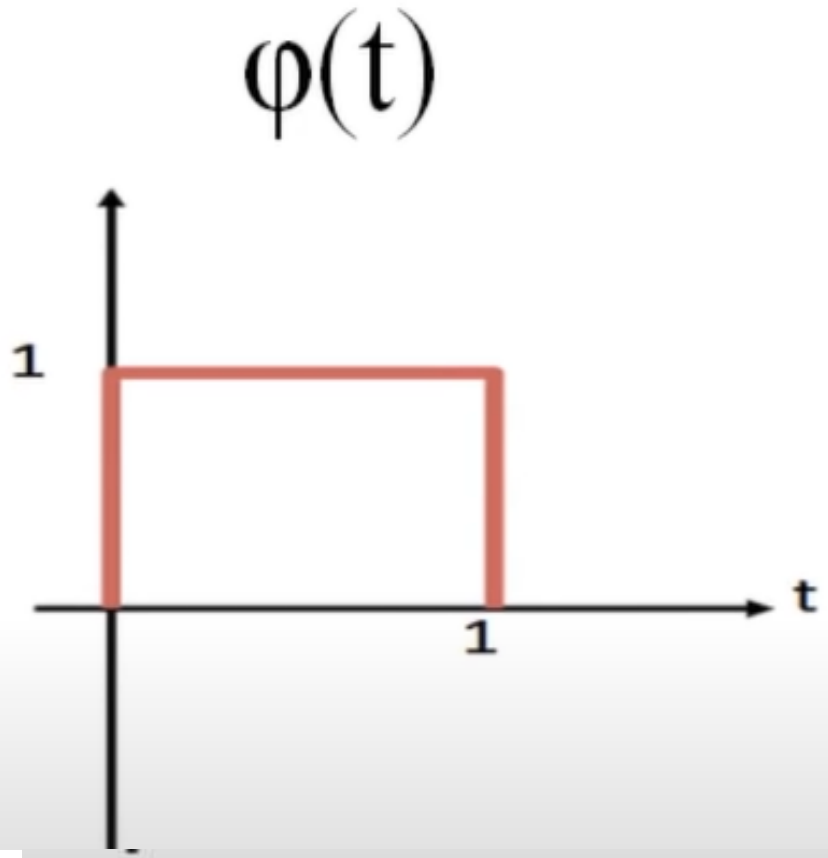
Haar Transform

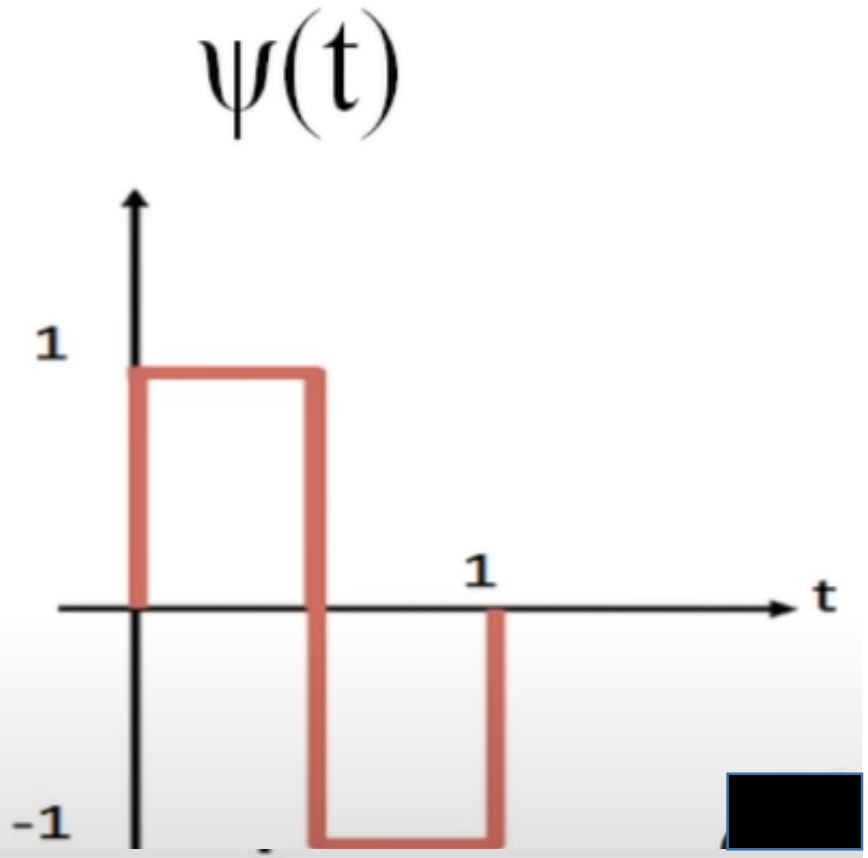
- The Haar transform, also known as the Haar wavelet transform, is a mathematical technique used in signal processing and image compression. It is named after the Hungarian mathematician Alfréd Haar, who developed it in the early 20th century.

- The Haar transform is based on a class of orthogonal matrices whose elements are either 1, -1 or 0 multiplied by powers of $\sqrt{2}$.

- The Haar transform is computationally efficient transform as the transform of an N-point vector requires only 2(N-1) additions and N multiplications.

- Haar scaling and wavelet functions:





computes average (low pass)

computes details (high pass)

Walsh Transform

- The Fourier analysis is basically the representation of a signal by a set of orthogonal sinusoidal waveforms.
(perpendicular)
- The coefficients of this representation are called frequency components and the waveforms are ordered by frequency.
- Walsh in 1923 introduced a complete set of orthonormal square wave functions to represent these functions.
- The computational simplicity of the Walsh function is due to the fact that Walsh functions are real and they take only two values which are either +1 or -1.

1-D Walsh Transform

This transform is slightly different from the others

Suppose we have a function $f(x), x = 0, \ldots, N-1,$

where $N = 2^n$.

We use binary representation for x and u.

We need n bits to represent them.

$$x_{10} = \left(b_{n-1}(x) \ldots b_0(x)\right)_2$$

Example: 1-D Walsh Transform

Suppose $f(x)$ has $N = 8$ samples.

In that case $n = 3$ since $N = 2^n$.

Consider $f(6)$:

$x_{10} = 6 \Rightarrow x_2 = 110 \Rightarrow b_0(6) = 0, b_1(6) = 1, b_2(6) = 1$

1-D Walsh Transform

Kernel values- controls all the major functions of the hardwar

- We define now the 1-D Walsh transform as follows:

$$W(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) \left[\prod_{i=0}^{n-1} (-1)^{b_i(x) b_{n-1-i}(u)} \right]$$

- The above is equivalent to:

$$W(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) (-1)^{\sum_{i=1}^{n-1} b_i(x) b_{n-1-i}(u)}$$

- The transform kernel values are obtained from:

$$T(u,x) = T(x,u) = \frac{1}{N} \left[\prod_{i=0}^{n-1} (-1)^{b_i(x) b_{n-1-i}(u)} \right] = \frac{1}{N} (-1)^{\sum_{i=1}^{n-1} b_i(x) b_{n-1-i}(u)}$$

- Therefore, the array formed by the Walsh matrix is a real symmetric matrix. It is easily shown that it has orthogonal columns and rows.

1-D Walsh Transform

- We would like to write the Walsh transform in matrix form.
- We define the vectors

$$\underline{f} = \begin{bmatrix} f(0) & f(1) & \dots & f(N-1) \end{bmatrix}^T$$

$$\underline{W} = \begin{bmatrix} W(0) & W(1) & \dots & W(N-1) \end{bmatrix}^T$$

- The Walsh transform can be written in matrix form

$$\underline{W} = T \cdot \underline{f}$$

- As mentioned in previously, matrix T is a real, symmetric matrix with orthogonal columns and rows. We can easily show that it is unitary and therefore:

$$T^{-1} = N \cdot T^T = N \cdot T, \; N \text{ is the size of the signal}$$

1-D Inverse Walsh Transform

- Base on the last equation of the previous slide we can show that the Inverse Walsh transform is almost identical to the forward transform!

$$f(x) = \sum_{x=0}^{N-1} W(u) \left[\prod_{i=0}^{n-1} (-1)^{b_i(x) b_{n-1-i}(u)} \right]$$

- The above is again equivalent to

$$f(x) = \sum_{x=0}^{N-1} W(u) (-1)^{\sum_{i=1}^{n-1} b_i(x) b_{n-1-i}(u)}$$

- The array formed by the inverse Walsh matrix is identical to the one formed by the forward Walsh matrix apart from a multiplicative factor N.

2-D Walsh Transform

- We define now the 2-D Walsh transform as a straightforward extension of the 1-D transform:

$$W(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v)}\right]$$

2-D Inverse Walsh Transform

- We define now the Inverse 2-D Walsh transform. It is identical to the forward 2-D Walsh transform!

$$f(x,y) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} W(u,v)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v)}\right]$$

Implementation

- The 2-D Walsh transform is separable and symmetric.

- Therefore it can be implemented as a sequence of two 1-D Walsh transforms, in a fashion similar to that of the 2-D DFT.

Basis Functions of Walsh Transform

- Remember that the Fourier transform is based on trigonometric terms.
- The Walsh transform consists of basis functions whose values are only 1 and -1.
- They have the form of square waves.
- These functions can be implemented more efficiently in a digital environment than the exponential basis functions of the Fourier transform.

Kernels of Forward and Inverse Walsh Transform

- For 1-D signals the forward and inverse Walsh kernels differ only in a constant multiplicative factor of N .
- This is because the array formed by the kernels is a symmetric matrix having orthogonal rows and columns, so its inverse array is the same as the array itself!
- In 2-D signals the forward and inverse Walsh kernels are identical!

The Concept of Sequency

- The concept of frequency exists also in Walsh transform basis functions.

- We can think of frequency as the number of zero crossings or the number of transitions in a basis vector and we call this number **sequency**.

Computation of the Walsh Transform

- For the fast computation of the Walsh transform there exists an algorithm called **Fast Walsh Transform (FWT).**

- This is a straightforward modification of the FFT.

Hadamard Transform

- Hadamard Transform is basically the same as Walsh Transform except the rows of the transform matrix are re-ordered.
- The elements of mutually orthogonal basis vectors of a Hadamard transform are either +1 or -1.
- It results in very low computational complexity in the calculation of the transform coefficients.

- **Technique used in signal processing, image analysis, and digital communication.**

2D Hadamard Transform

- It is similar to the 2-D Walsh transform.
- 2-D Hadamard transform is given by:

$$H(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_i(u)+b_i(y)b_i(v)}\right]$$

or

$$H(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)(-1)^{\sum_{i=1}^{n-1}(b_i(x)b_i(u)+b_i(x)b_i(u))}$$

2D Inverse Hadamard Transform

- It is identical to the forward 2-D Hadamard transform.
- 2D Inverse Hadamard transform is given by:

$$f(x,y) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} H(u,v)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_i(u)+b_i(y)b_i(v)}\right]$$

or

$$f(x,y) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} H(u,v)(-1)^{\sum_{i=1}^{n-1}(b_i(x)b_i(u)+b_i(x)b_i(u))}$$

Properties of Hadamard Transform

- Most of the properties of Walsh transform are valid with Hadamard Transform.
- The Hadamard transform differs from the Walsh transform only in the order of basis functions.
- The order of basis functions of the Hadamard transform does not allow the fast computation of it by using a straightforward modification of the FFT.

Recursive Relationship of the Hadamard Transform

- H_N represents Hadamard matrix of order N as follows:

$$H_{2N} = \begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix}$$

- Therefore, starting from a small Hadamard matrix, we can compute Hadamard matrix of any size.

- This is good reason to use Hadamard Transform.

Ordered Walsh and Hadamard Transforms

- **Ordered transforms:**

- Modified versions of the Walsh and Hadamard transforms formed by rearranging the rows of the transformation matrix.

- The sequency increases as the index of the transform increases.

Ordered Walsh and Hadamard Transforms

- The ordered Walsh/Hadamard transforms exhibit the property of energy compaction whereas the original versions of the transforms do not.

- Among all the transforms of this family, the Ordered Hadamard is the most popular due to recursive matrix property and energy compaction.

Summary

- Similarity with Walsh Transform.
- Computational efficiency.
- Due to recursive relationship of Hadamard transform, it is not desirable to store the entire matrix.

Slant Transform

- Slant transform was introduced by Enomoto and Shibata as an orthogonal transform containing discrete sawtooth waveforms or 'slant' basis vectors.
- Specifically designed for image coding.
- A slant basis vector that is monotonically decreasing in constant steps from maximum to minimum has the sequency property and has a fast computational algorithm.

Slant Transform (Cont.)

- A slant basis vector efficiently represents linear brightness variations along an image line.

- The slant transformation has been utilized in several transform image-coding systems for monochrome and color images.

Slant Transform (Cont.)

- Let S_N denote an N x N slant matrix with $N = 2^n$. Then,

$$S_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- The S_4 matrix is obtained by the following operation:

$$S_4 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ a & b & -a & b \\ 0 & 1 & 0 & -1 \\ -b & a & b & a \end{bmatrix} \begin{bmatrix} S_2 & 0 \\ 0 & S_2 \end{bmatrix}$$

Slant Transform (Cont.)

- If a = 2b and b = $\frac{1}{\sqrt{5}}$, the slant matrix is given by:

$$S_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \frac{3}{\sqrt{5}} & \frac{1}{\sqrt{5}} & -\frac{1}{\sqrt{5}} & -\frac{3}{\sqrt{5}} \\ 1 & -1 & -1 & 1 \\ \frac{1}{\sqrt{5}} & -\frac{3}{\sqrt{5}} & \frac{3}{\sqrt{5}} & -\frac{1}{\sqrt{5}} \end{bmatrix}$$

Slant Transform (Cont.)

- The sequency of the matrix of order four is given by:

$$S_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \frac{3}{\sqrt{5}} & \frac{1}{\sqrt{5}} & -\frac{1}{\sqrt{5}} & -\frac{3}{\sqrt{5}} \\ 1 & -1 & -1 & 1 \\ \frac{1}{\sqrt{5}} & -\frac{3}{\sqrt{5}} & \frac{3}{\sqrt{5}} & -\frac{1}{\sqrt{5}} \end{bmatrix}$$

Zero Sign Change

One Sign Change

Two Sign Change

Three Sign Change

Slant Transform (Cont.)

- The slant transform reproduces linear variations of brightness very well.
- However its performance at edges is not as optimal as the KL transform or Discrete Cosine Transform.
- Because of the 'slant' nature of the lower order coefficients, its effect is to smear the edges.

WHAT IS IMAGE RESTORATION?

- Image restoration attempts to restore images that have been degraded
  - Identify the degradation process and attempt to reverse it
  - Similar to image enhancement, but more objective

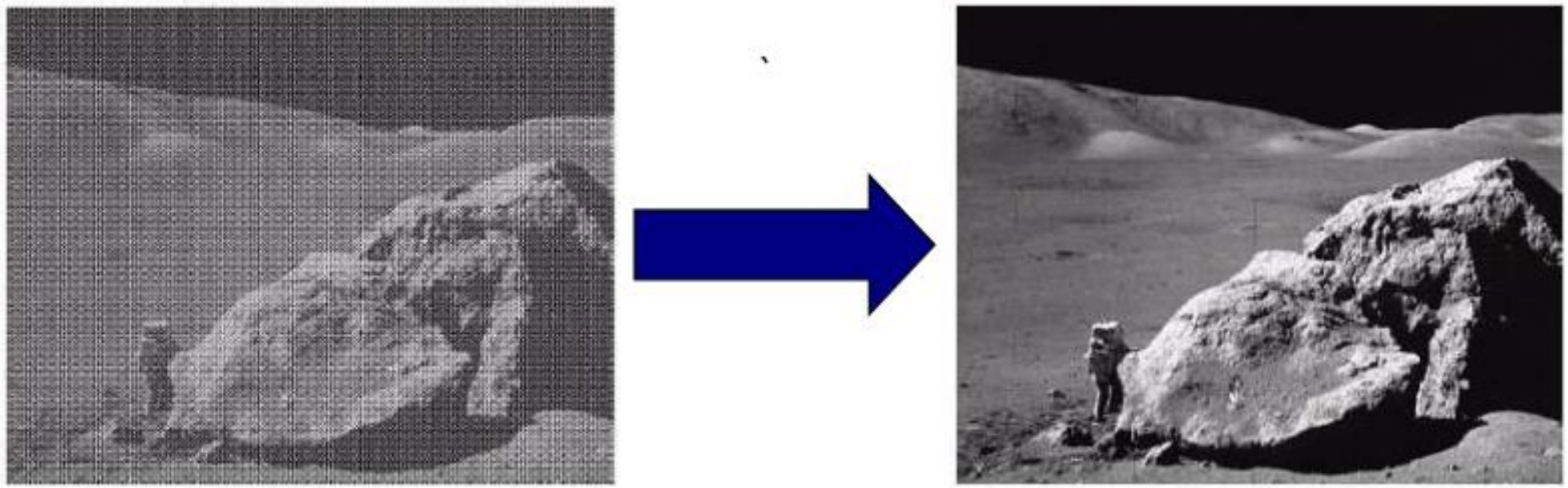


IMAGE RESTORATION

- The sources of noise in digital images arise during image acquisition (digitization) and transmission
  - Imaging sensors can be affected by ambient conditions
  - Interference can be added to an image during transmission
  - Relative motion between camera and object which causes motion blur.
    - Gauss Blur
    - Out-of Focus Blur
    - Motion Blur

IMAGE RESTORATION

```
close all;
clear all;
clc;
a = imread(horse.jpg)
a = rgb2gray(a);
H = fspecial(motion,
MotionBlur_a = imfilte
imshow(a),
title(Original Image)
imshow(MotionBlur a),
title(Motion Blurred
```

Fig. (*a*) *Original image* (*b*) *Motion-blurred image with* [10 25] (*c*) *Motion-blurred image with* [15 35] (*d*) *Motion-blurred image with* [25 50]

CLASSIFICATION OF IMAGE RESTORATION TECHNIQUES

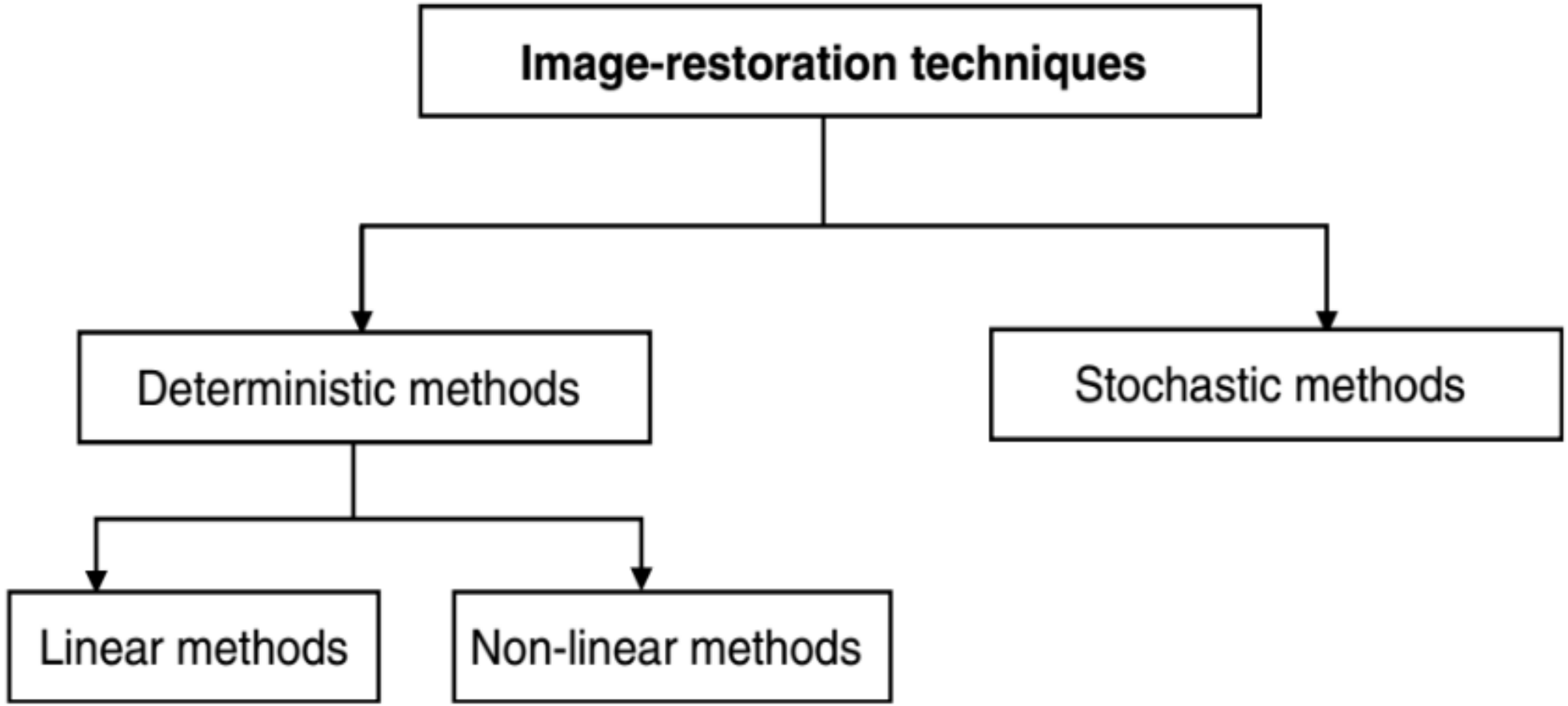
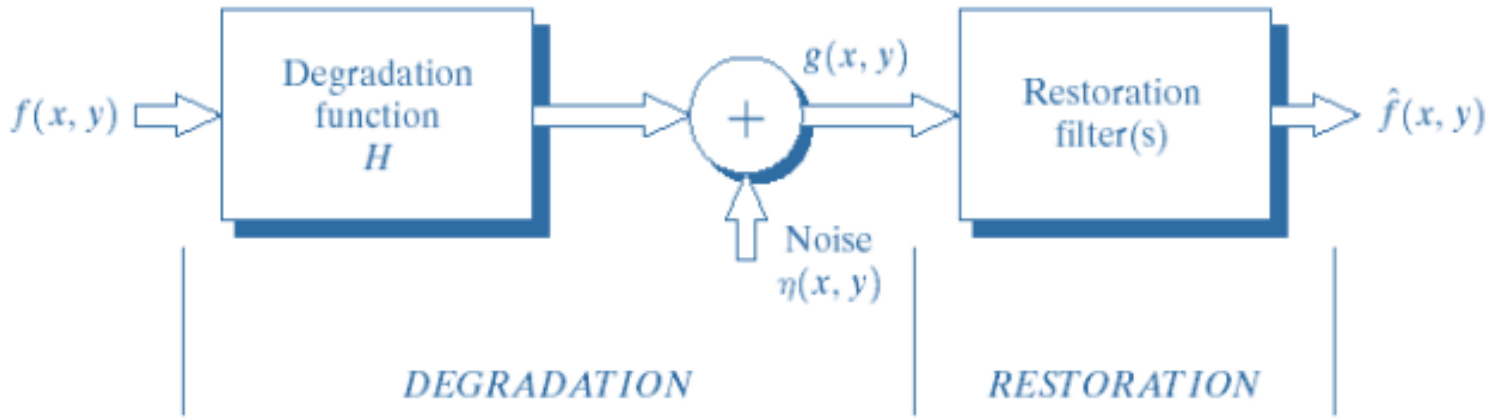


Fig. *Classification of image-restoration techniques*

IMAGE RESTORATION MODEL



- We model the degradation process by a degradation function h(x,y), an additive noise term, η(x,y), as

  g(x,y)=h(x,y)∗f(x,y)+ η(x,y)
  - f(x,y) is the (input) image free from any degradation
  - g(x,y) is the degraded image
  - ∗ is the convolution operator
- The goal is to obtain an estimate of f(x,y) according to the knowledge about the degradation function h and the additive noise η

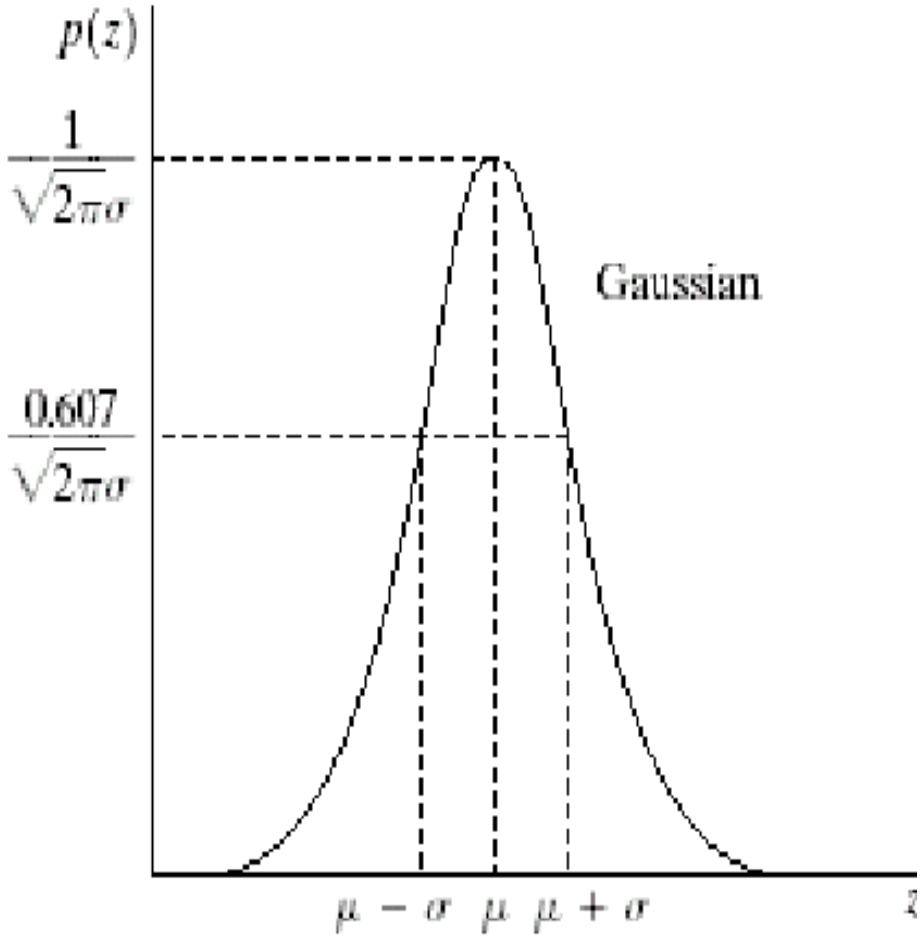
  In frequency domain: G(u,v)=H(u,v)F(u,v)+N(u,v)

NOISE MODELS-GAUSSIAN NOISE

•Noise (image) can be classified according the distribution of the values of pixels (of the noise image) or its (normalized) histogram

•Gaussian noise is characterized by two parameters, μ (mean) and σ^2 (variance), by

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(z-\mu)^2/2\sigma^2}$$



•70% values of z fall in the range [(μ-σ),(μ+σ)]

•95% values of z fall in the range [(μ-2σ),(μ+2σ)]

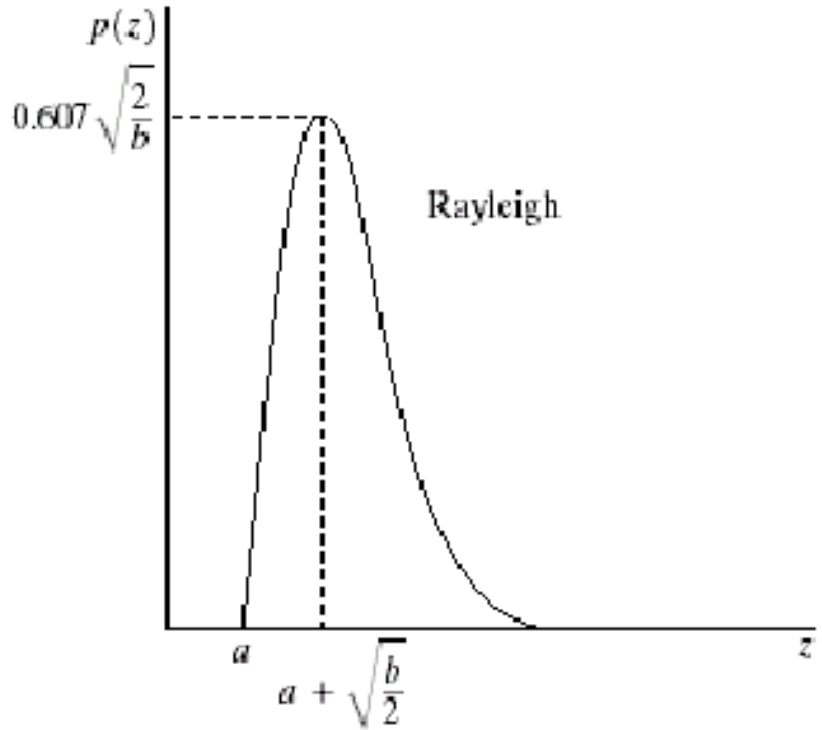
NOISE MODELS-RAYLEIGH NOISE

$$p(z) = \begin{cases} \frac{2}{b}(z-a)e^{-(z-a)^2/b} & \text{for } z \geq a \\ 0 & \text{for } z < a \end{cases}$$

The mean and variance of this density are given by

$$\mu = a + \sqrt{\pi b / 4} \quad \text{and} \quad \sigma^2 = \frac{b(4-\pi)}{4}$$

a and b can be obtained through mean and variance



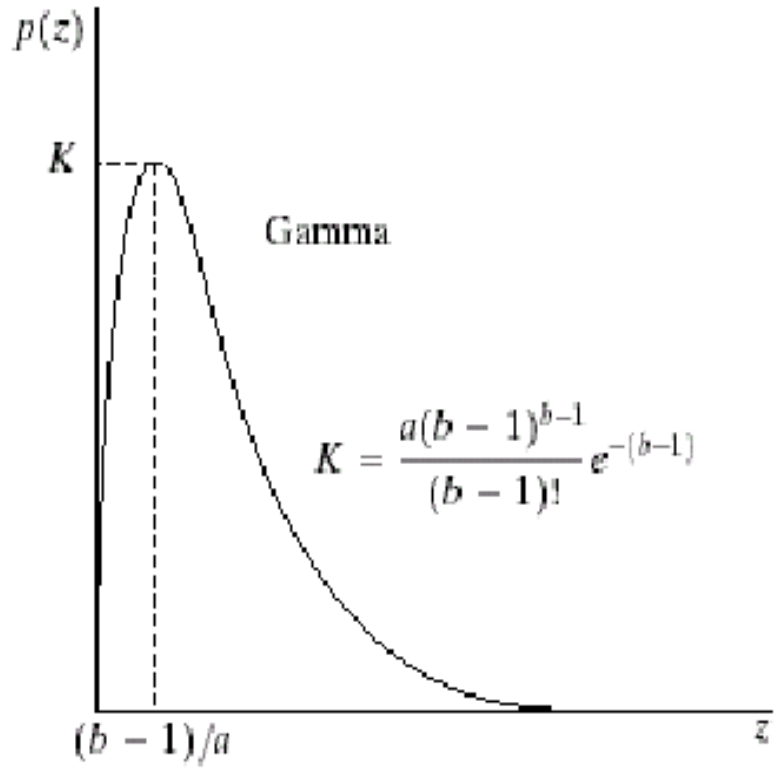
NOISE MODELS-ERLONG (Gamma) NOISE

$$p(z) = \begin{cases} \dfrac{a^b z^{b-1}}{(b-1)!} e^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}$$

The mean and variance of this density are given by

$$\mu = b/a \text{ and } \sigma^2 = \frac{b}{a^2}$$

a and b can be obtained through mean and variance



NOISE MODELS-EXPONENTIAL NOISE

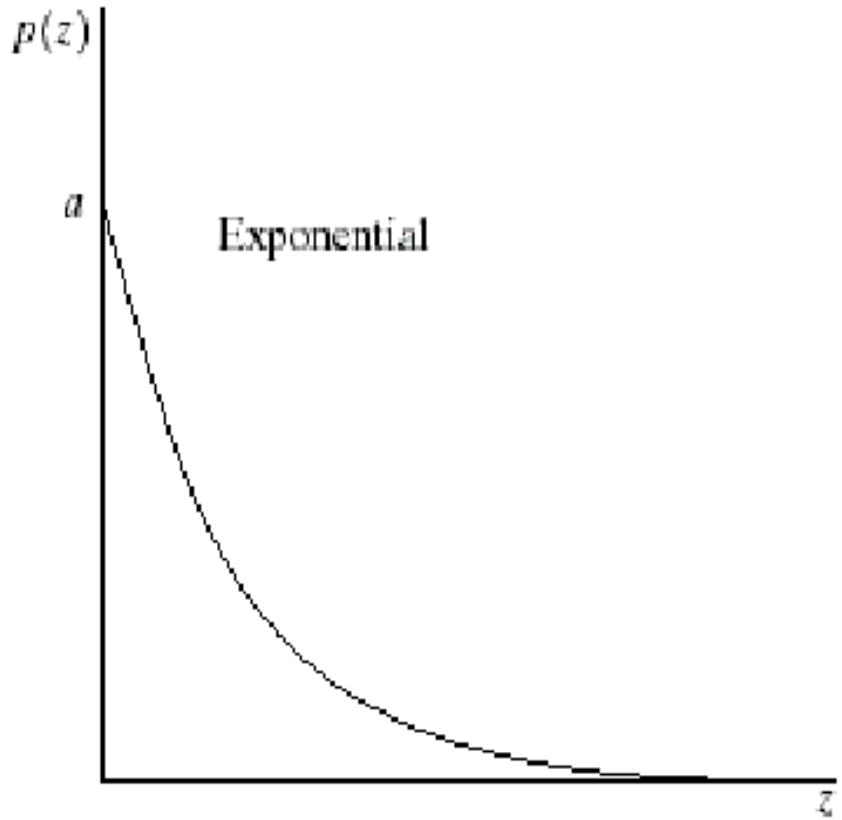
- **Exponential** noise

$$p(z) = \begin{cases} ae^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}$$

  - The mean and variance of this density are given by

$$\mu = 1/a \text{ and } \sigma^2 = \frac{1}{a^2}$$

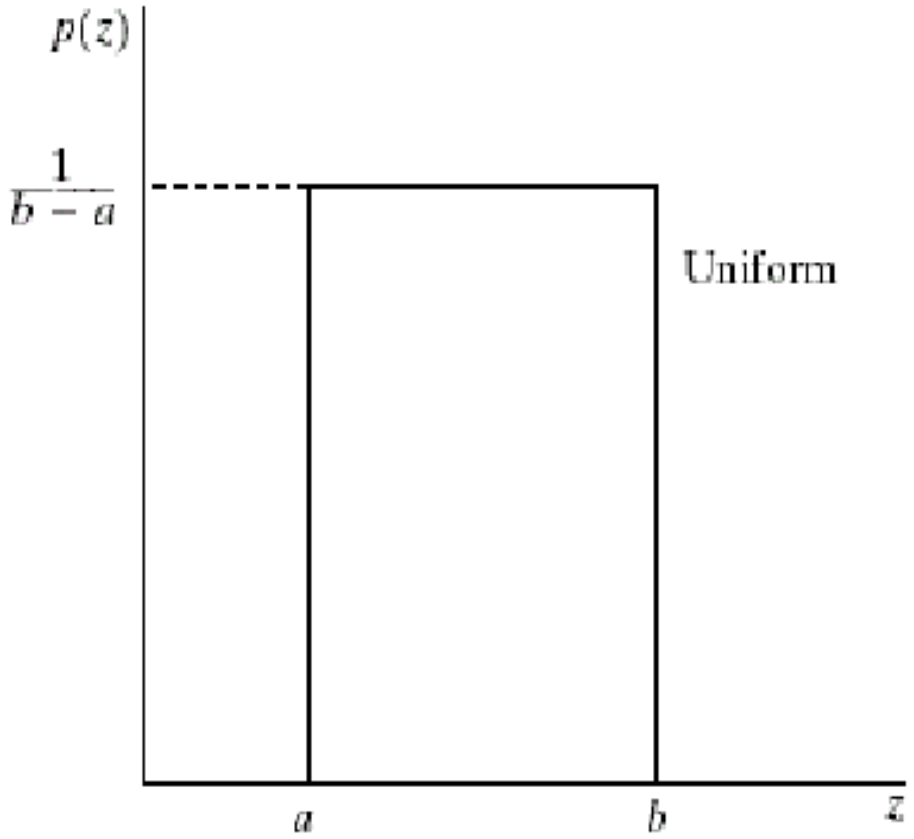
  - Special case pf Erlang PDF with b=1



NOISE MODELS-UNIFORM NOISE

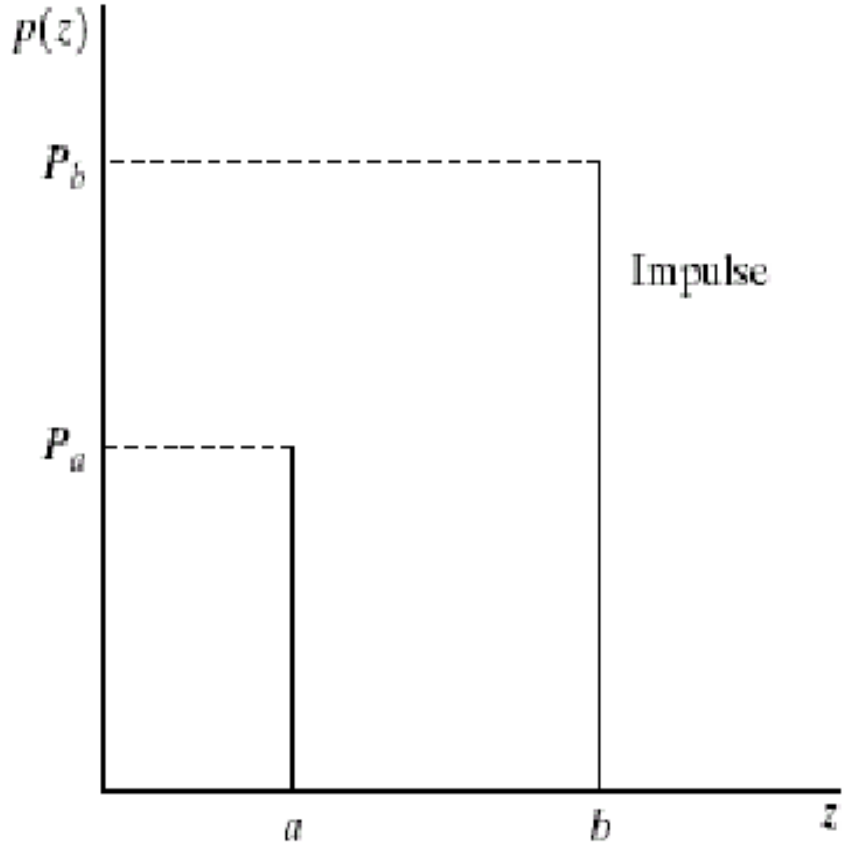
- **Uniform** noise
  - $$p(z) = \begin{cases} \dfrac{1}{b-a} & \text{if } a \le z \le b \\ 0 & \text{otherwise} \end{cases}$$
  - The mean and variance of this density are given by

$$\mu = (a+b)/2 \text{ and } \sigma^2 = \frac{(b-a)^2}{12}$$



NOISE MODELS-SALT & PEPPER NOISE

- **Impulse** (salt-and-pepper) noise
  - $$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$$
  - If either Pa or Pb is zero, the impulse noise is called unipolar
  - a and b usually are extreme values because impulse corruption is usually large compared with the strength of the image signal
  - It is the only type of noise that can be distinguished from others visually



NOISE REMOVAL

We can use spatial filters of different kinds to remove different kinds of noise

The *arithmetic mean* filter is a very simple one and is calculated as follows:

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

This is implemented as the simple smoothing filter

It blurs the image.

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t)\in S_{xy}} g(s,t)$$

There are different kinds of mean filters all of which exhibit slightly different behaviour:

- Geometric Mean
- Harmonic Mean
- Contraharmonic Mean

AVERAGE FILTERS

Geometric Mean:

Achieves similar smoothing to the arithmetic mean, but tends to lose less image detail.

$$\hat{f}(x,y) = \left[\prod_{(s,t)\in S_{xy}} g(s,t) \right]^{\frac{1}{mn}}$$

Harmonic Filter:

Works well for salt noise, but fails for pepper noise.

Also does well for other kinds of noise such as Gaussian noise.

$$\hat{f}(x,y) = \frac{mn}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

Contraharmonic Mean:

Q is the *order* of the filter and adjusting its value changes the filter's behaviour

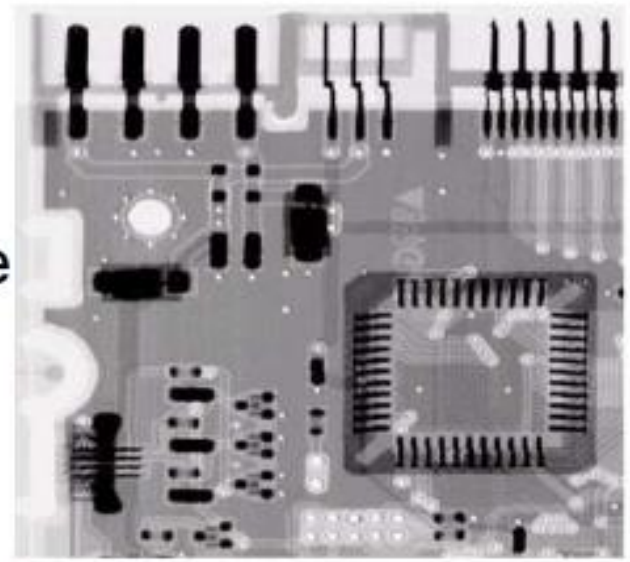
Positive values of Q eliminate pepper noise

Negative values of Q eliminate salt noise

$$\hat{f}(x,y) = \frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

NOISE REMOVAL

Original image



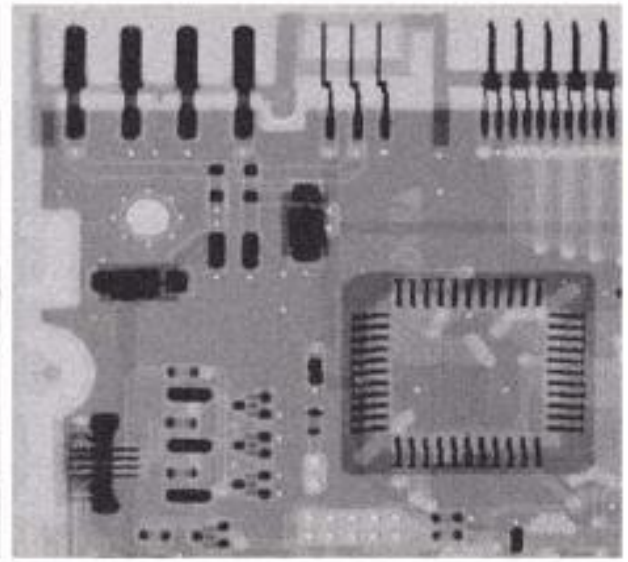
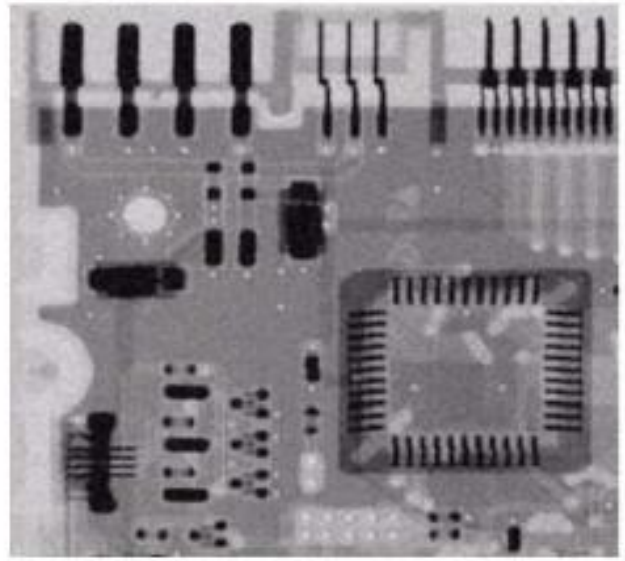
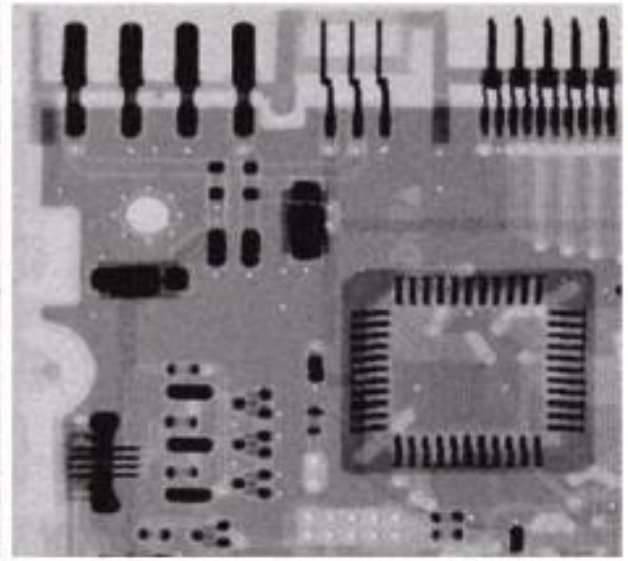


Image corrupted by Gaussian noise

3x3 Arithmetic Mean Filter





3x3 Geometric Mean Filter (less blurring than AMF, the image is sharper)

NOISE REMOVAL

Image corrupted by pepper noise at 0.1

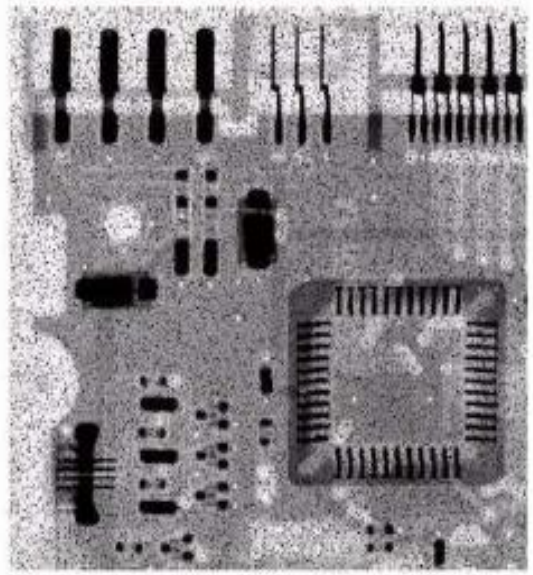
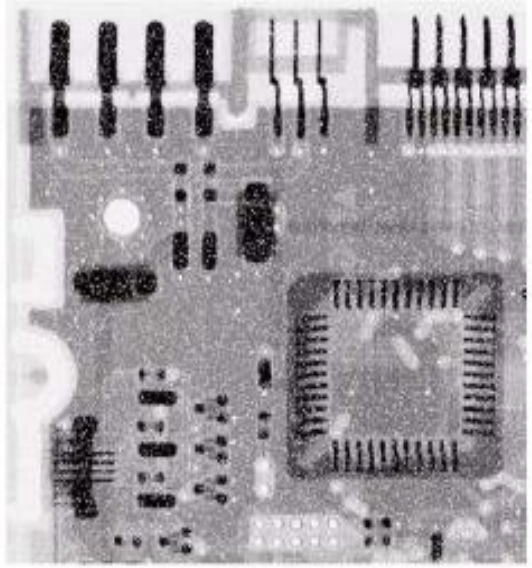
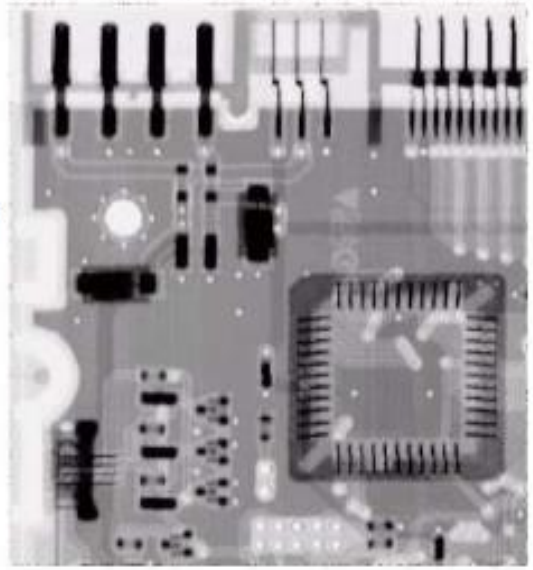


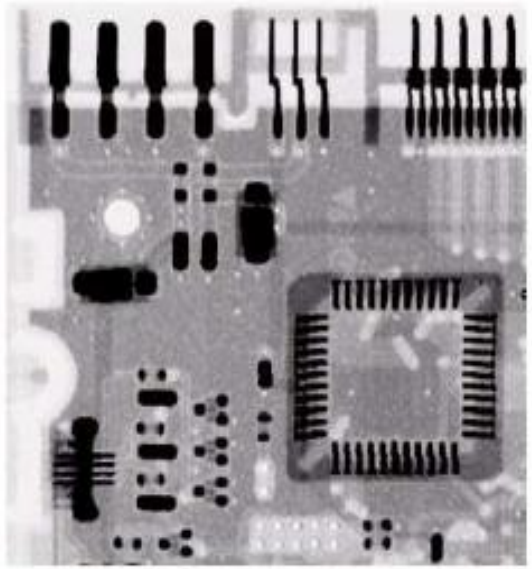
Image corrupted by salt noise at 0.1



Filtering with a 3x3 Contraharmonic Filter with Q=1.5

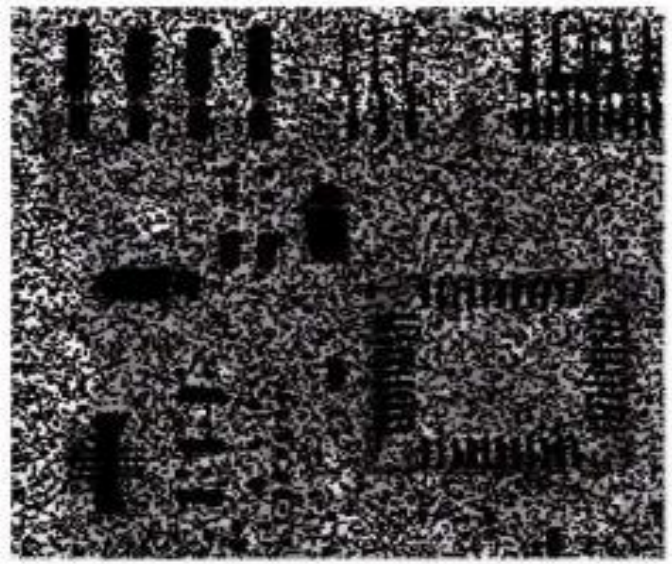


Filtering with a 3x3 Contraharmonic Filter with Q=-1.5

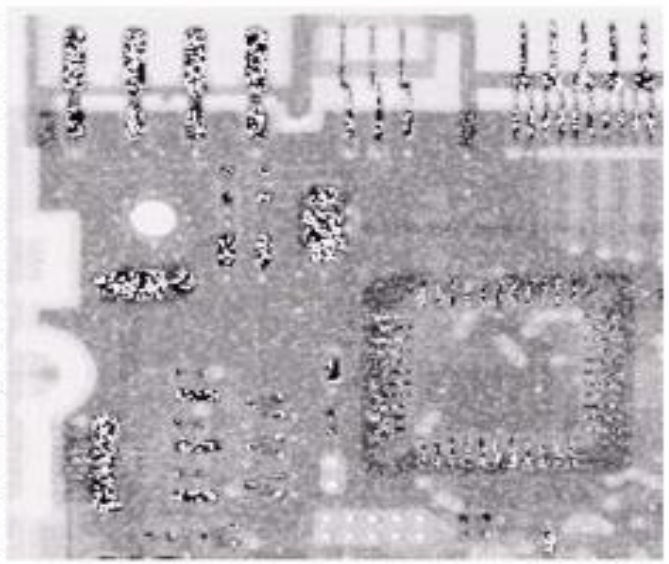


NOISE REMOVAL

Choosing the wrong value for Q when using the contraharmonic filter can have drastic results



Pepper noise filtered by a 3x3 CF with Q=-1.5



Salt noise filtered by a 3x3 CF with Q=1.5

NOISE REMOVAL

Spatial filters based on ordering the pixel values that make up the neighbourhood defined by the filter support.

Useful spatial filters include

- Median filter
- Max and min filter
- Midpoint filter
- Alpha trimmed mean filter

NOISE REMOVAL

Median Filter:

$$\hat{f}(x,y) = \underset{(s,t)\in S_{xy}}{median}\{g(s,t)\}$$

Excellent at noise removal, without the smoothing effects that can occur with other smoothing filters.

Particularly good when salt and pepper noise is present.

Max Filter:

$$\hat{f}(x,y) = \max_{(s,t)\in S_{xy}} \{g(s,t)\}$$

Min Filter:

$$\hat{f}(x,y) = \min_{(s,t)\in S_{xy}} \{g(s,t)\}$$

Max filter is good for pepper noise and Min filter is good for salt noise.

Alpha-Trimmed Mean Filter:

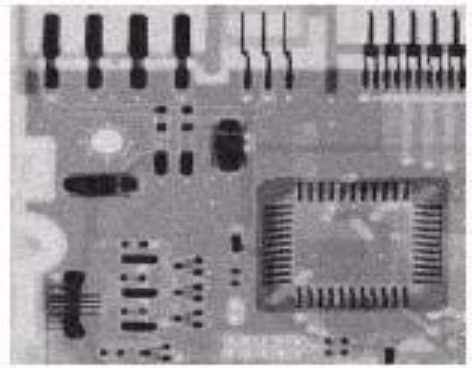
$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t)$$

We can delete the $d/2$ lowest and $d/2$ highest grey levels.

So $g_r(s, t)$ represents the remaining $mn - d$ pixels.

NOISE REMOVAL

Image corrupted by uniform noise



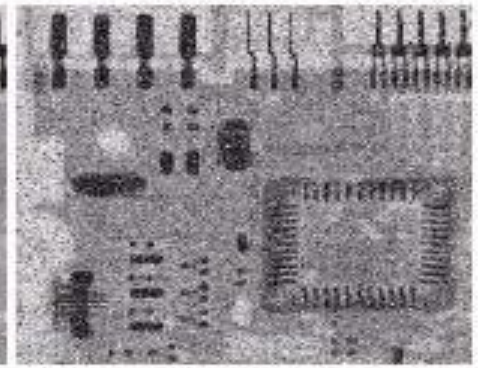
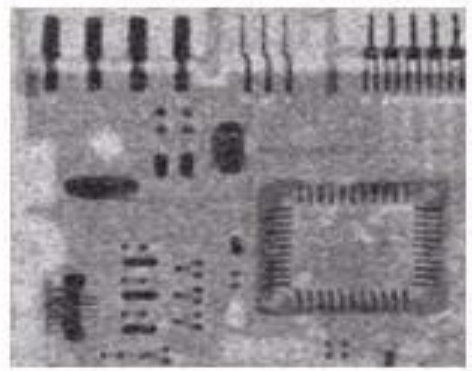
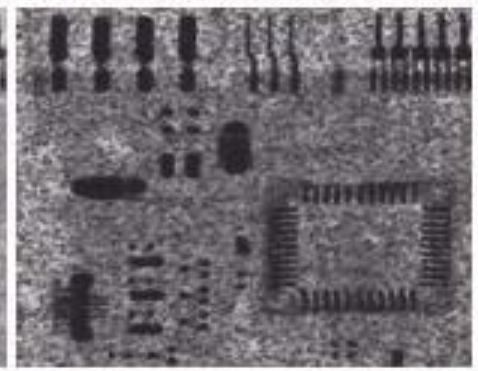


Image further corrupted by Salt and Pepper noise

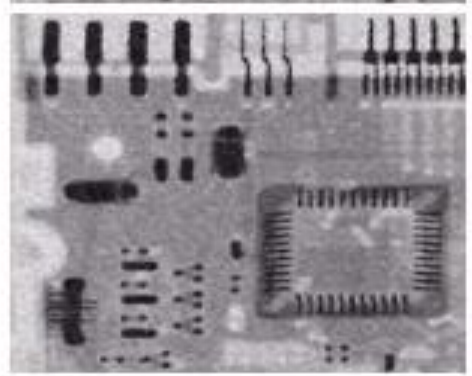
Filtering by a 5x5 Arithmetic Mean Filter

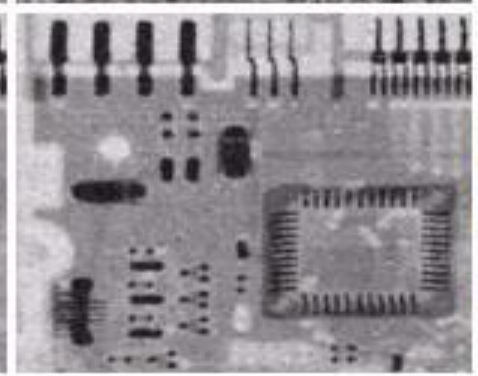




Filtering by a 5x5 Geometric Mean Filter

Filtering by a 5x5 Median Filter



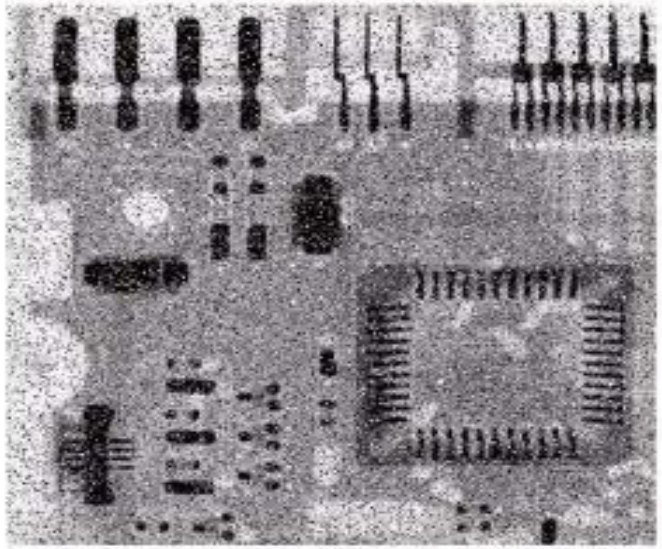


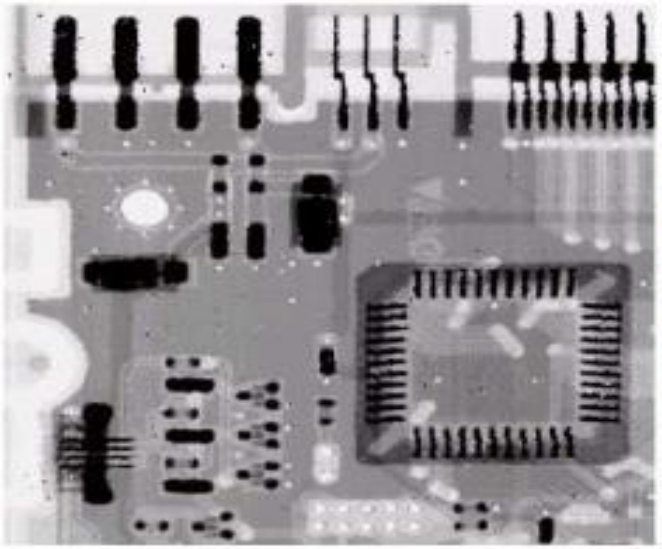
Filtering by a 5x5 Alpha-Trimmed Mean Filter (d=5)

NOISE REMOVAL

Repeated passes remove the noise better but also blur the image

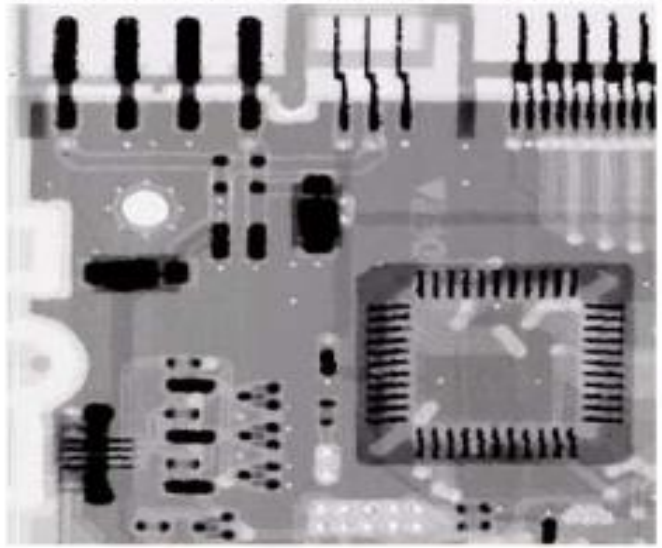
Image corrupted by Salt And Pepper noise at 0.2

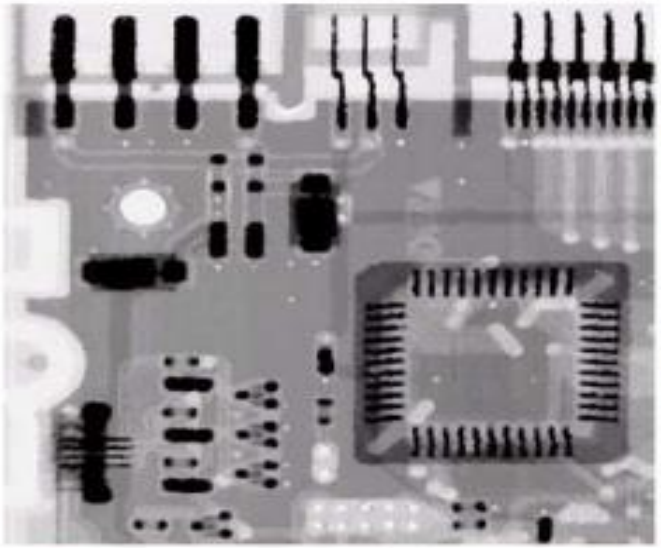




Result of 1 pass with a 3x3 Median Filter

Result of 2 passes with a 3x3 Median Filter

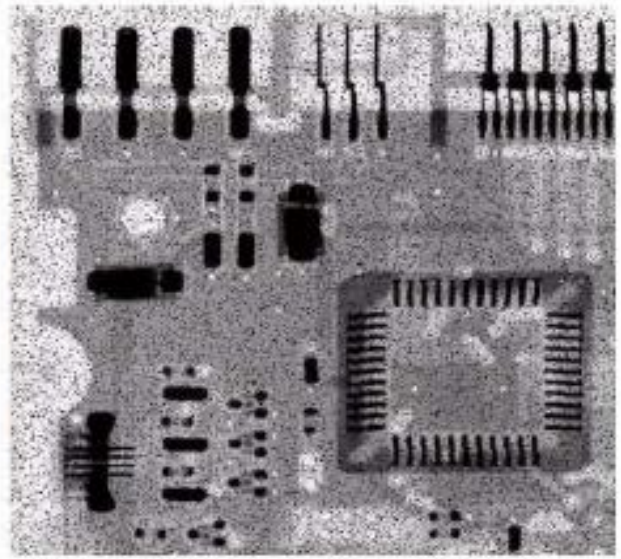




Result of 3 passes with a 3*3 Median Filter

NOISE REMOVAL

Image corrupted by Pepper noise



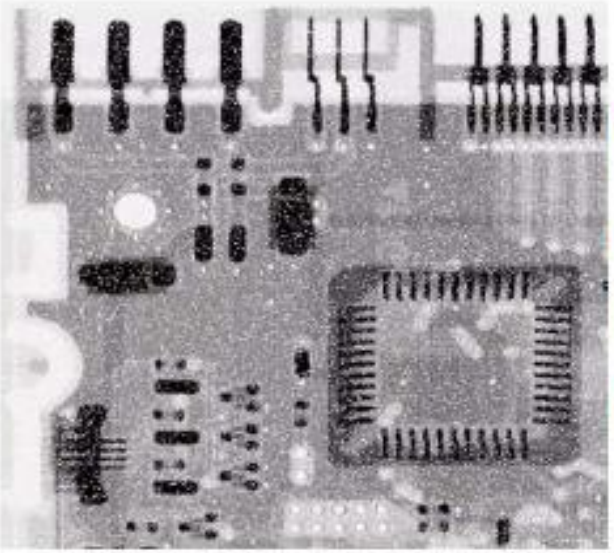
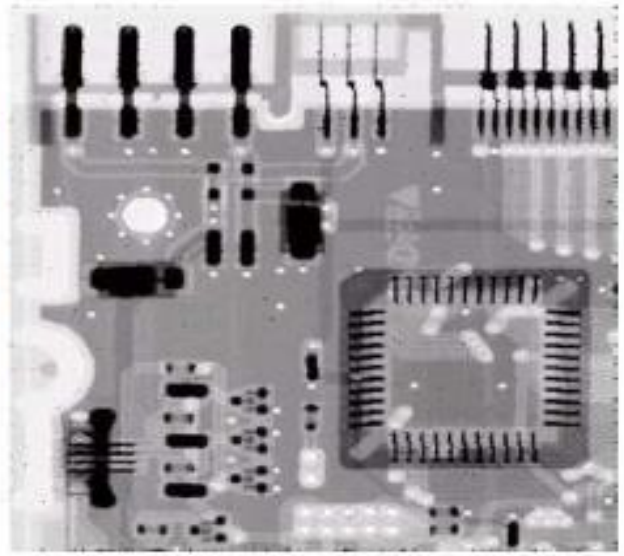
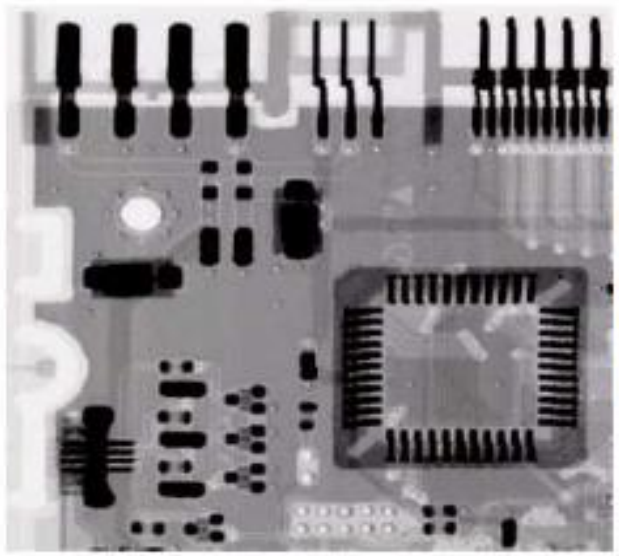


Image corrupted by Salt noise

Filtering above with a 3x3 Max Filter





Filtering above with a 3x3 Min Filter