Syllabus

Introduction to Signal Processing - Linear Algebra for Signal Processing – Complex Bases for Real Signals – Convolution – From DFT to FFT- Z Domain Representation of Signals – Digital Filter Design- Elements of digital image processing - Image model - Sampling and quantization - Relationships between pixels - Image Transforms - Discrete Fourier Transform, Discrete Cosine Transform, Discrete Wavelet Transform –Image Enhancement: Enhancement by point processing - Spatial filtering - Enhancement in the frequency domain - Color Image Processing - Morphological Image Processing: Dilation and Erosion - Opening and Closing - Some basic morphological algorithms. Image Segmentation Region based, edge based, clustering based- Representation and Description - GLCM HOG, SIFT.

Introduction to Signals

Signals

- A signal is a function of one or more variables that conveys information about some (usually physical) phenomenon.
- For a function f , in the expression $f(t1, t2, . . . , tn)$, each of the $\{tk\}$ is called an independent variable, while the function value itself is referred to as a dependent variable.
- Some examples of signals include:
  - a voltage or current in an electronic circuit
  - the position, velocity, or acceleration of an object
  - a force or torque in a mechanical system
  - a flow rate of a liquid or gas in a chemical process
  - a digital image, digital video, or digital audio
  - a stock market index

- Continuous Signal x(t): Signal that are defined for every instance of time.
- Digital Signal x(n) : Signal that are defined at discrete instance of time.

$$x(nT) = x(t)\,\big|_{t=nT}$$

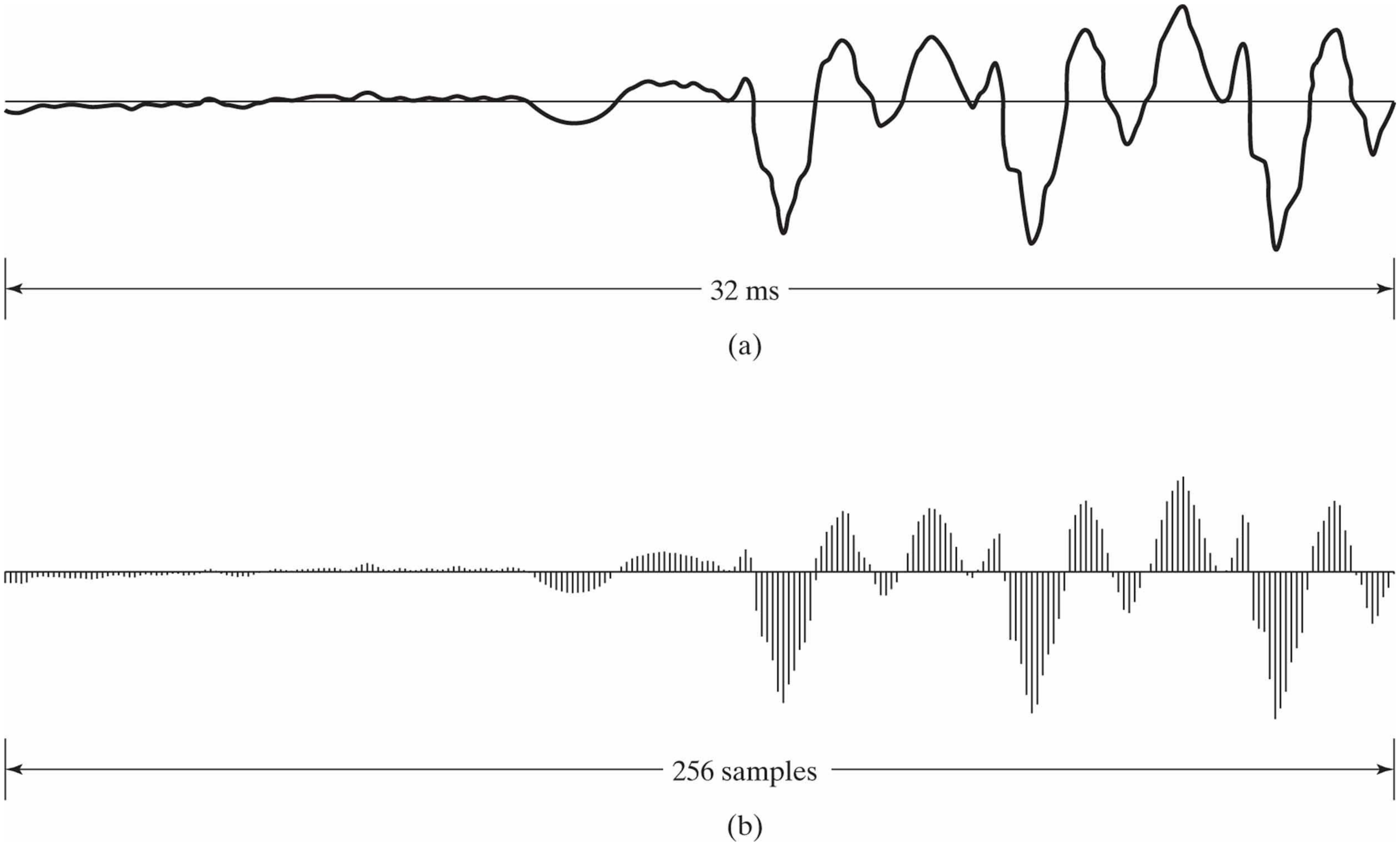
T= sampling period

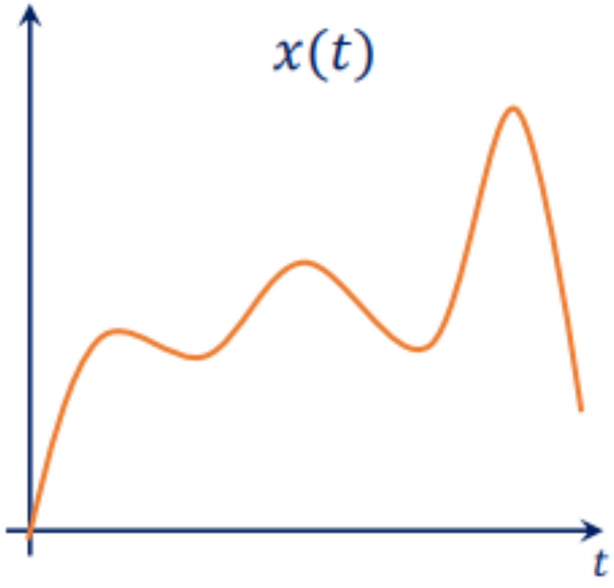
N=range

$$x(nT) = x(n); n = 0, \pm 1, \pm 2, \ldots.$$

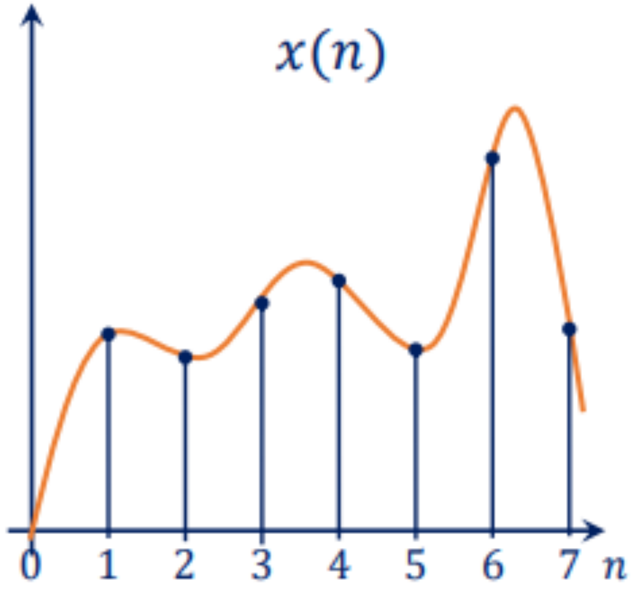
$$\ldots\ldots x(-2), x(-1), x(0), x(1), x(2), \ldots.$$

Figure 2.2 (a) Segment of a continuous-time speech signal $x_a(t)$. (b) Sequence of samples $x[n] = x_a(nT)$ obtained from the signal in part (a) with $T = 125\ \mu s$.

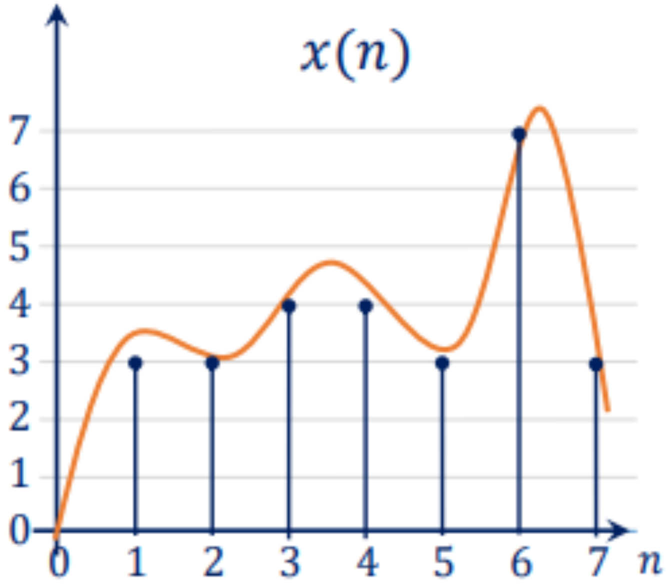




Continuous Signal



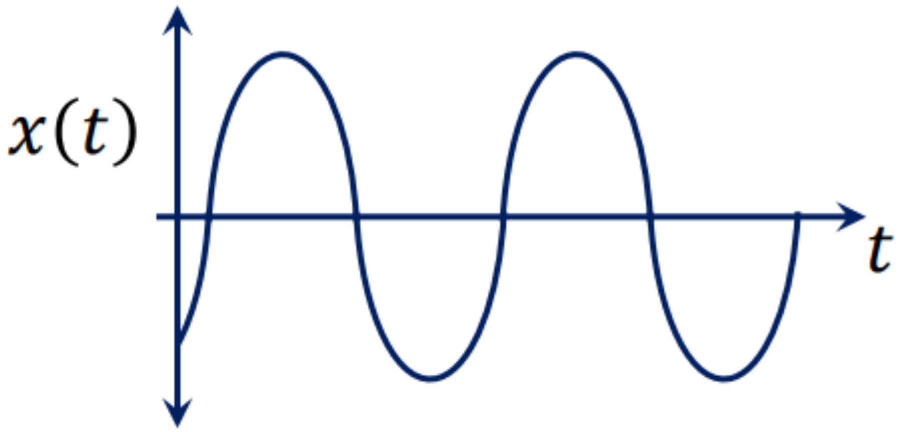
Discrete Time Signal



Digital Signal

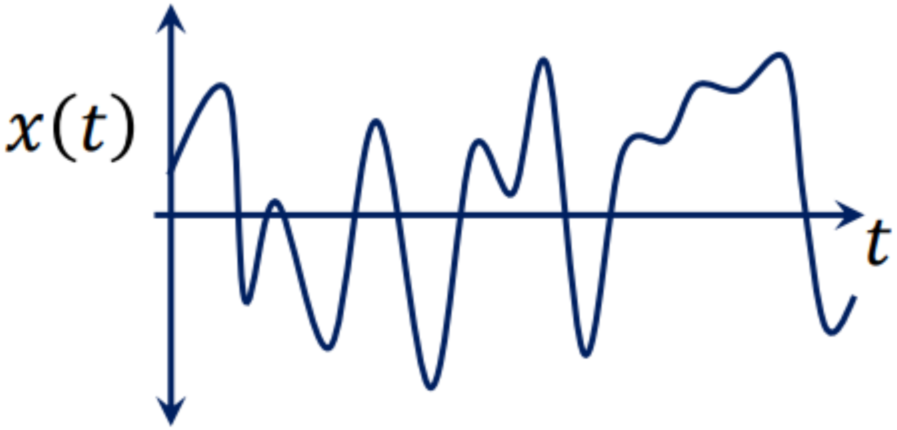
Deterministic Signal

- A signal is classified as deterministic, if its magnitude can be calculated for any point of time. Deterministic signals can be expressed using mathematical function of independent variable. For example, $x\;t = A \sin(\omega t + \varphi\;)$.



Random Signal

- A signal whose value cannot be predicted at any point of time is classified as random signal.



$eg : x(n) = 2.e^{-2t}$, with T=0.2, Interval 0 ≤ t ≤ 2. Find x(t)

Analog

X(0)=2; X(0.2)=1.34; X(0.4)=0.89;
X(0.6)=0.60;X(0.8)=0.40;X(1)=0.27
;X(1.2)=0.18;X(1.4)=0.12,..

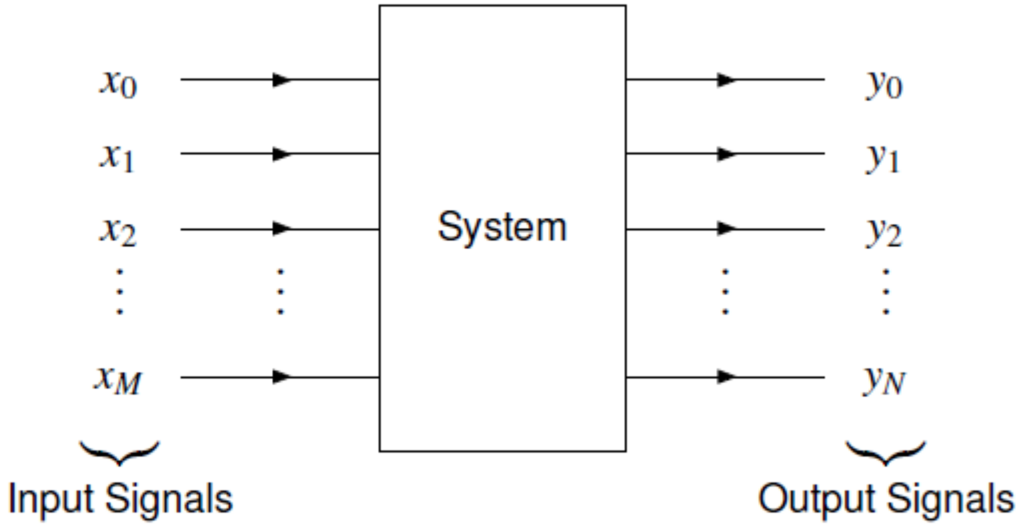
Digital

$$x(nT) = x(t)\,|_{t=nT}$$
$$= x(0.2n)$$
$$x(n) = 2e^{-2(0.2n)}$$
$$= 2.e^{-0.4n}$$

X(0)=2;x(1)=1.34;x(2)=0.89;x(3)=0.60;
x(4)=0.40;x(5)=0.27

System

- Physical device that generate response or an output signal for a given input signal



Y(t)=T[x(t)] Continuous time system
Y(n)=T[x(n)] Discrete time system

- Determine the response of the following system to the I/P signal

$$x(n) = \begin{cases} n, & -3 \le n \le 3 \\ 0 & n>3, n=0, n<3 \end{cases}$$

1. y(n)=x(n)
2. y(n)=x(n-1)
3. y(n)=x(n+1)

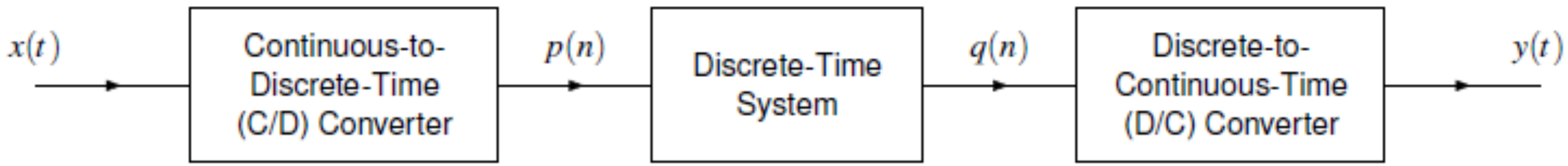
Classification of Systems

- Number of inputs:
  - A system with one input is said to be single input (SI).
  - A system with more than one input is said to be multiple input (MI).
- Number of outputs:
  - A system with one output is said to be single output (SO).
  - A system with more than one output is said to be multiple output (MO).
- Types of signals processed:
  - A system can be classified in terms of the types of signals that it processes.
  - Consequently, terms such as the following (which describe signals) can also be used to describe systems:
    - one-dimensional and multi-dimensional,
    - continuous-time (CT) and discrete-time (DT), and
    - analog and digital.
- For example, a continuous-time (CT) system processes CT signals and a discrete-time (DT) system processes DT signals.

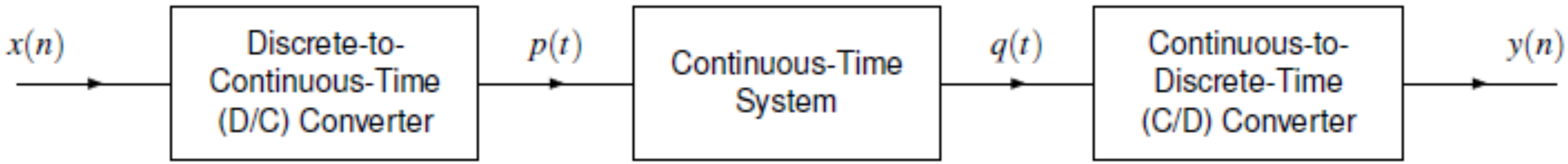
A CT signal is called a function ;

A DT signal is called a sequence

Signal Processing Systems



Processing a Continuous-Time Signal With a Discrete-Time System



Processing a Discrete-Time Signal With a Continuous-Time System

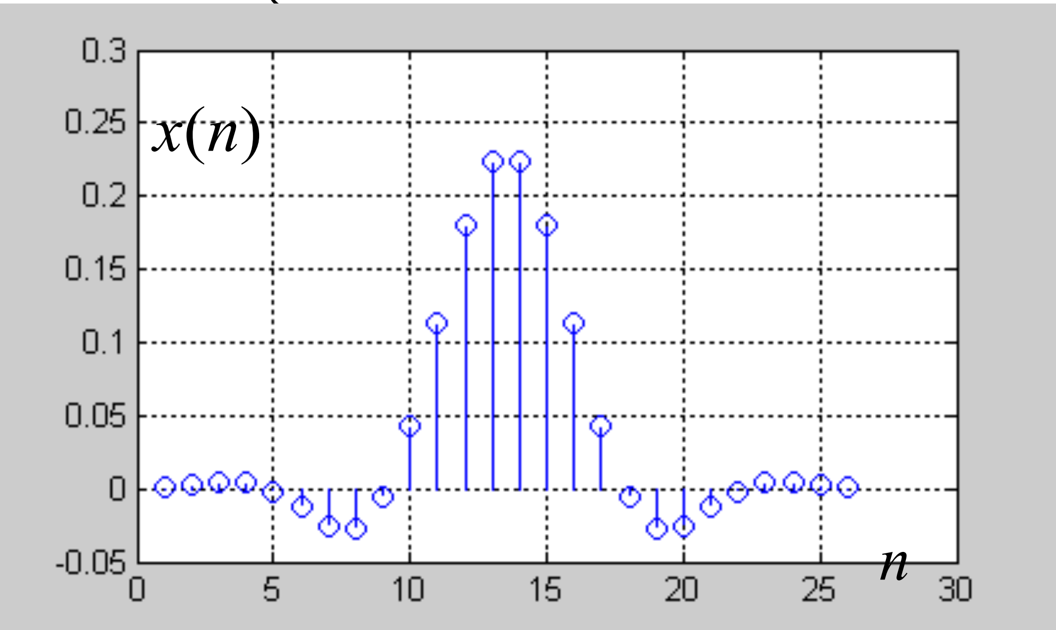
Discrete Time Signal

- A DTS x(n) is a function of an independent variable that is an integer.

A. Functional representation:

$$x(n) = \begin{cases} 1 & for \quad n = 1,3 \\ 6 & for \quad n = 0,7 \\ 0 & elsewhere \end{cases} \qquad y(n) = \begin{cases} 0 & for \quad n < 0 \\ 0,6^n & for \quad n = 0,1,\ldots,102 \\ 1 & \quad n > 102 \end{cases}$$

B. Graphical representation



C. Tabular representation:

n	…	-2	-1	0	1	2
$x(n)$	…	0.12	2.01	1.78	5.23	0.12

D. Sequence representation:

$$x(n) = \{\ldots \quad 0.12 \quad 2.01 \quad 1.78 \quad 5.23 \quad 0.12 \quad \ldots\}$$

An infinite duration signal with the origin (n=0) indicate by ↑

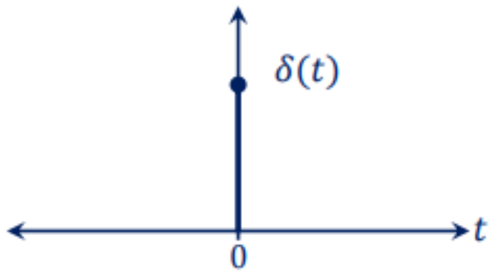
$$x(n) = \{\ldots\ldots\ldots, 0, \underset{\uparrow}{0}, 1, 2, 1, 0, 0, \ldots\ldots\ldots\ldots\ldots\}$$

Elementary Discrete-Time Signals

A. Unit sample sequence (unit sample, unit impulse, unit impulse signal)

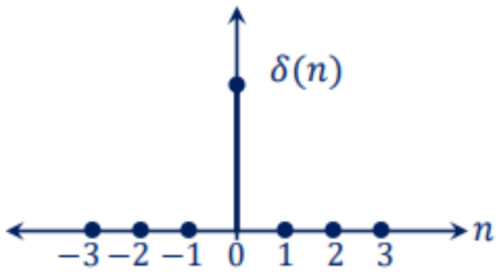
Continuous-Time Signal

$$\delta(t) = \begin{cases} 1 & t = 0 \\ 0 & t \neq 0 \end{cases}$$



Discrete-Time Signal

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}$$

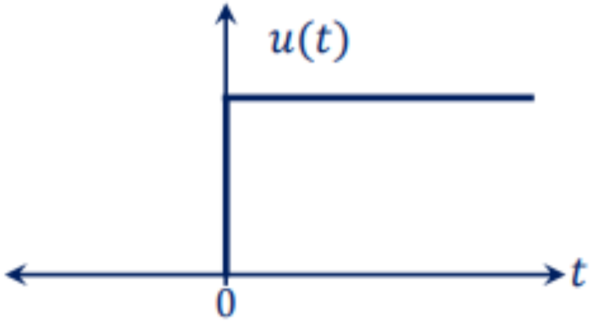


An impulse signal has zero value except at t = 0. It has infinitely high value t = 0

B. Unit step signal (unit step, Heaviside step sequence)

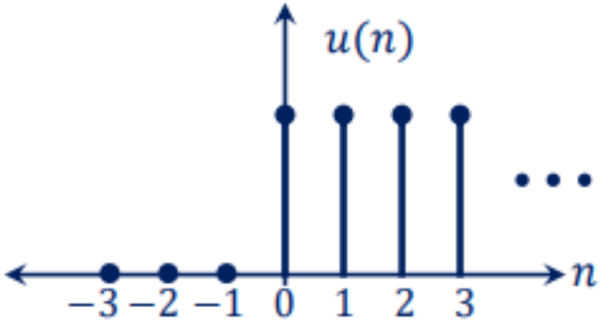
Continuous-Time Signal

$$u(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases}$$



Discrete-Time Signal

$$u(n) = \begin{cases} 1 & n \geq 0 \\ 0 & n < 0 \end{cases}$$

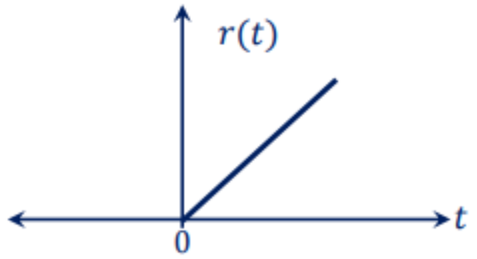


A unit step signal has unity value for $t \geq 0$ else zero value.

C. Unit Ramp Signal Ur(n)

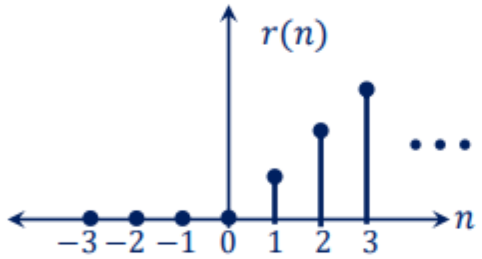
Continuous-Time Signal

$$r(t) = \begin{cases} t & t \geq 0 \\ 0 & t < 0 \end{cases}$$



Discrete-Time Signal

$$r(n) = \begin{cases} n & n \geq 0 \\ 0 & n < 0 \end{cases}$$

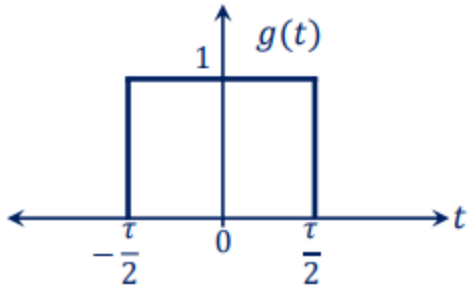


A ramp step signal has unity slop value for $t \geq 0$, otherwise it has zero value.

Rectangular Pulse Signal

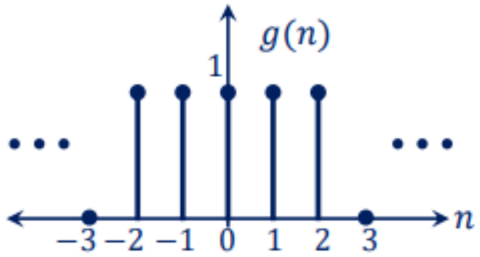
Continuous-Time Signal

$$g(t) = \begin{cases} 1 & -\frac{\tau}{2} \leq t \leq +\frac{\tau}{2} \\ 0 & \text{Otherwise} \end{cases}$$



Discrete-Time Signal

$$g(n) = \begin{cases} 1 & -m \leq n \leq +m \\ 0 & \text{Otherwise} \end{cases}$$



A unit rectangular pulse has unit amplitude within a time interval, otherwise it has zero value. It is also called the Gate pulse, Pulse function, or Window function, etc.

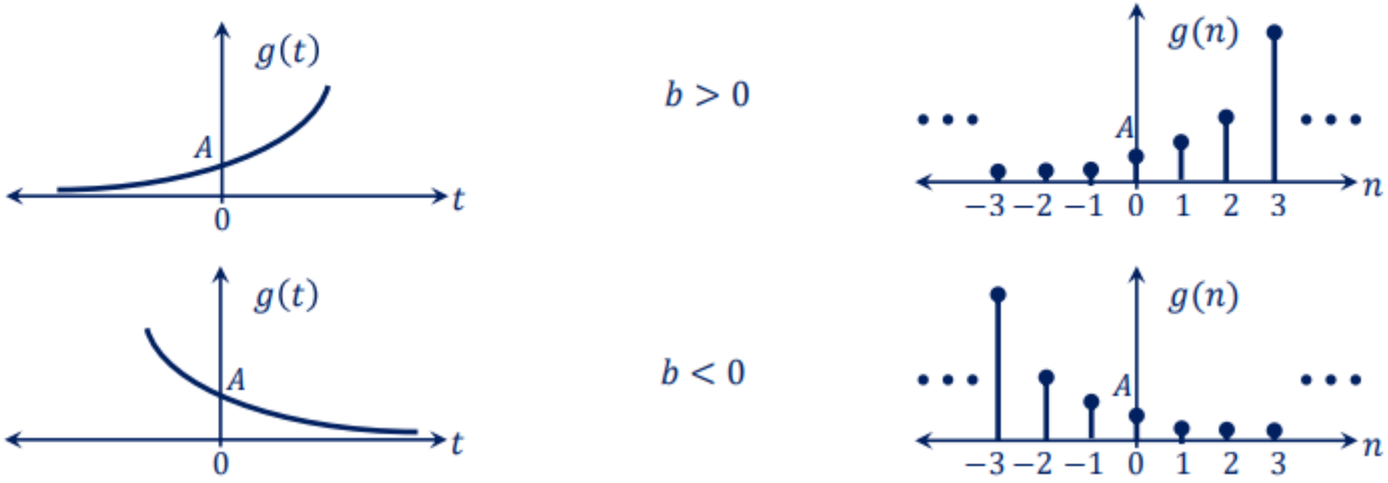
Exponential Signal

Continuous-Time Signal

$g(t) = Ae^{bt} \qquad A > 0$

Discrete-Time Signal

$g(n) = Ae^{bn} \qquad A > 0$



An exponential signal can either have exponentially rising or falling amplitude depending upon its exponent value.

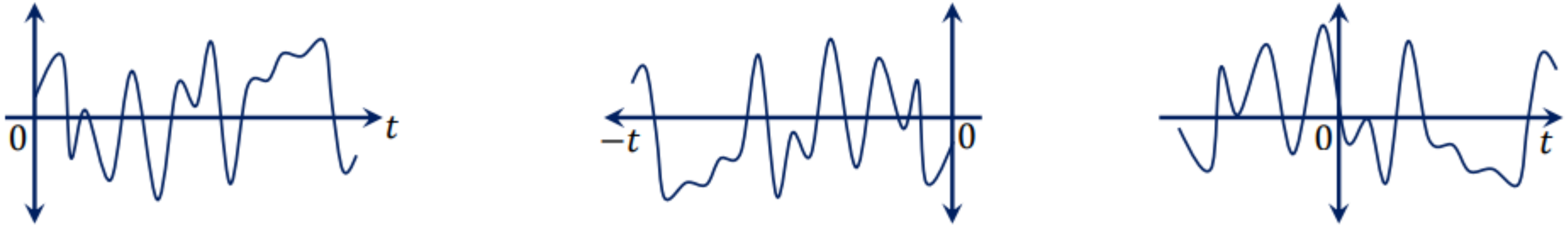
Causal, Anti-causal & Non-causal Signals

- Term causality is usually used to characterize systems. However, on the same lines signal can also be classified as causal, non-causal or anti-causal signals. Signal $x(t)$ is a

(a) Causal signal, if $|x(t)| > 0$ for $t \geq 0$.

(b) Anti-causal signal if $|x(t)| > 0$ for $t < 0$.

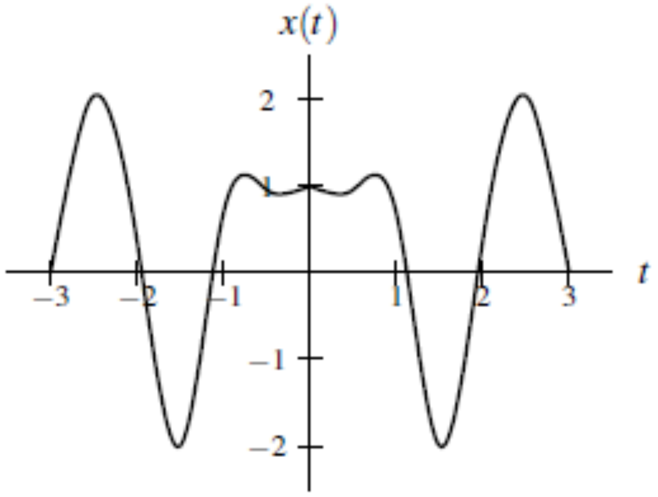
(c) Non-causal signal if $|x(t)| > 0$ for all $t \geq 0$ and $t < 0$.

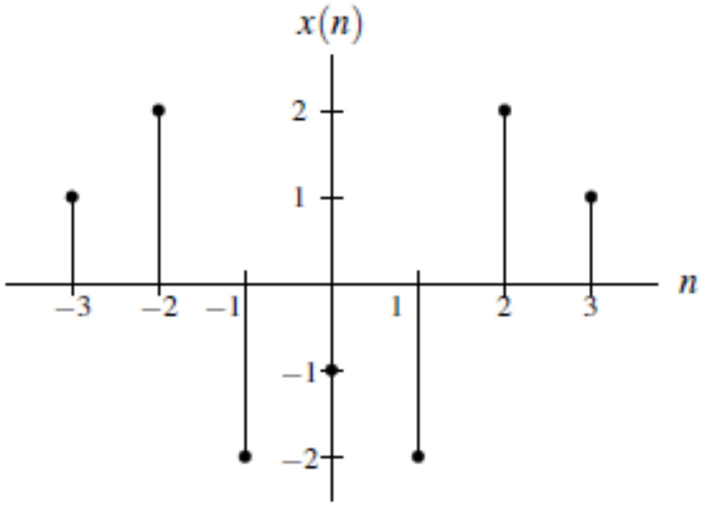


Causal Signal **Anti-Causal Signal** **Non-Causal Signal**

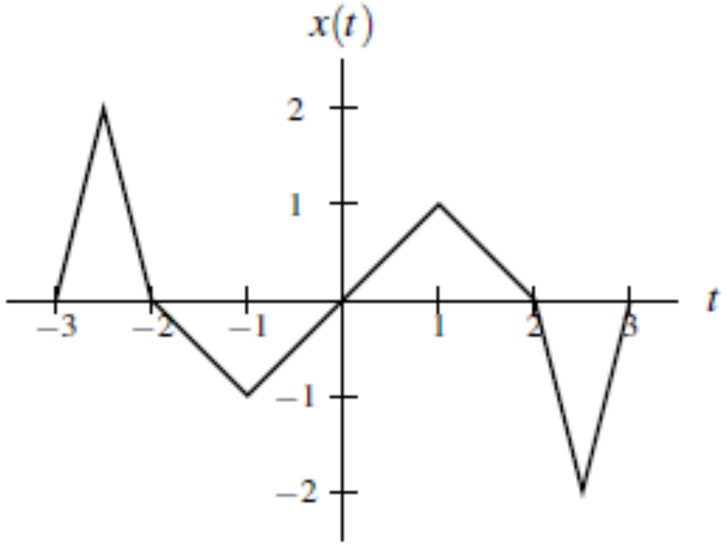
Odd and Even Signal

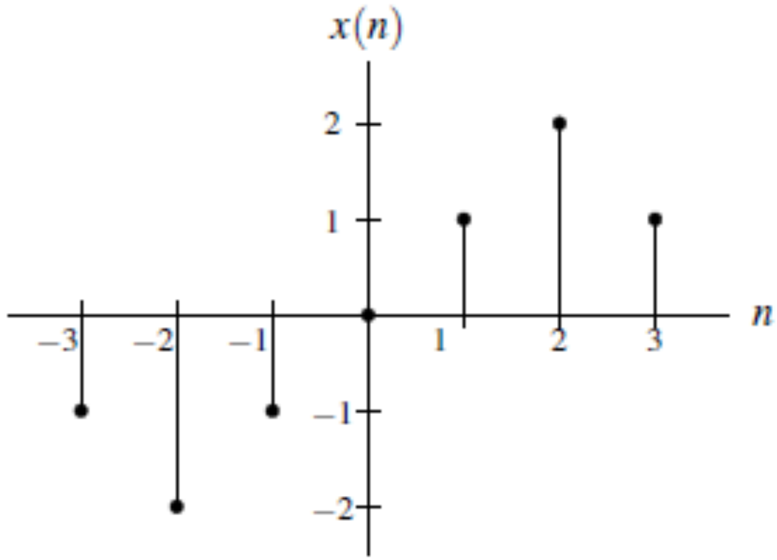
- Even Signal: A signal x(t) is said to be an even signal, if x(t) = x(-t)
- For Discrete Time even signal x(n) =x(-n)
- Geometrically, the graph of an even signal is symmetric about the origin.





- Odd Signal : The signal $x\ t$ is termed as an odd signal, if $x(t) = -x(-t)$
- For Discrete-Time odd signals $x(n) = -x(-n)$
- An odd signal x must be such that x(0) = 0.

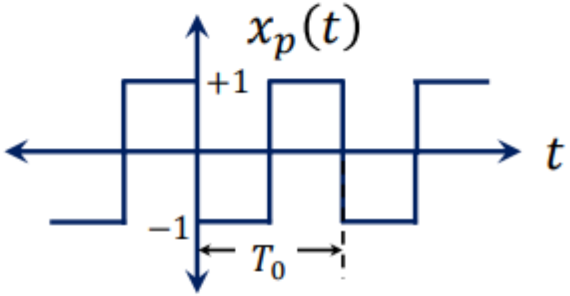




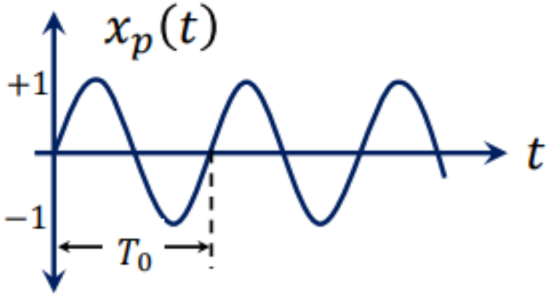
Periodic & Aperiodic Signals

- A signal $x(t)$ is said to be a periodic if it repeats after a finite time interval $T0$. Mathematically, $x(t) = x(t + T0)$. $T0$ is called fundamental time period and $f0 = 1/T0$ is called its fundamental frequency.

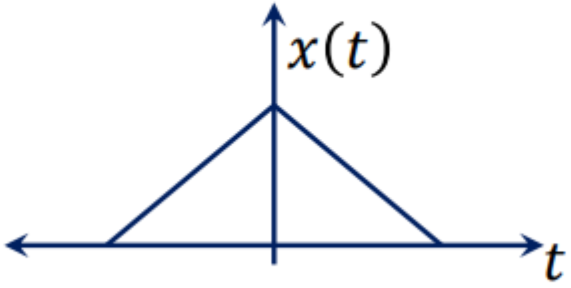
- An aperiodic or non-periodic signal repeats after infinity time, i.e. $T0 \rightarrow \infty$



Periodic



Periodic



Aperiodic

Classification of Discrete Time Systems

A discrete-time system is called **static** or **memoryless** if its output at any time instant n depends on the input sample at the same time, but not on the past or future samples of the input. In the other case, the system is said to be **dynamic** or to have **memory**.

If the output of a system at time n is completly determined by the input samples in the interval from n-N to n ($N \geq 0$), the system is said to have memory of **duration N**.

If $N = 0$, the system is **static** or **memoryless**.

If $0 < N < \infty$, the system is said to have **finite memory**.

If $N \to \infty$, the system is said to have **infinite memory**.

24

Examples:

The static (memoryless) systems:

$$y(n) = nx(n) + bx^3(n)$$

The dynamic systems with finite memory:

$$y(n) = \sum_{k=0}^{N} h(k)x(n-k)$$

The dynamic system with infinite memory:

$$y(n) = \sum_{k=0}^{\infty} h(k)x(n-k)$$

Time-Invariant vs. Time-Variable Systems

- A discrete-time system is called ***time-invariant*** if its input-output characteristics do not change with time. In the other case, the system is called ***time-variable***.
- **Definition.** A relaxed system $H[.]$ is ***time-*** or ***shift-invariant*** if only if

$$y(n) \equiv H\left[x(n)\right] \qquad x(n) \xrightarrow{H} y(n)$$

implies that

$$y(n-k) \equiv H\left[x(n-k)\right] \qquad x(n-k) \xrightarrow{H} y(n-k)$$

for ***every input signal*** $x(n)$ and ***every time shift*** k .

Examples:

The time-invariant systems:

$$y(n) = x(n) + bx^3(n)$$

$$y(n) = \sum_{k=0}^{N} h(k)x(n-k)$$

The time-variable systems:

$$y(n) = nx(n) + bx^3(n-1)$$

$$y(n) = \sum_{k=0}^{N} h^{N-n}(k)x(n-k)$$

Linear vs. Non-linear Systems

A discrete-time system is called ***linear*** if only if it satisfies the ***linear superposition principle***. In the other case, the system is called ***non-linear***.

Definition. A relaxed system $H[.]$ is ***linear*** if only if

$$H\left[a_1x_1(n)+a_2x_2(n)\right]=a_1H\left[x_1(n)\right]+a_2H\left[x_2(n)\right]$$

for any arbitrary input sequences $x_1(n)$ and $x_2(n)$, and any arbitrary constants a_1 and a_2.

The multiplicative (scaling) property of a linear system:

$$H\left[a_1x_1(n)\right]=a_1H\left[x_1(n)\right]$$

The additivity property of a linear system:

$$H\left[x_1(n)+x_2(n)\right]=H\left[x_1(n)\right]+H\left[x_2(n)\right]$$

Causal vs. Non-causal Systems

Definition. A system is said to be ***causal*** if the output of the system at any time n (i.e., $y(n)$) depends only on present and past inputs (i.e., $x(n)$, $x(n-1)$, $x(n-2)$, ...). In mathematical terms, the output of a ***causal*** system satisfies an equation of the form

$$y(n) = F\left[x(n), x(n-1), x(n-2), \cdots\right]$$

where $F[.]$ is some arbitrary function. If a system does not satisfy this definition, it is called ***non-causal***.

Examples:

The causal system:

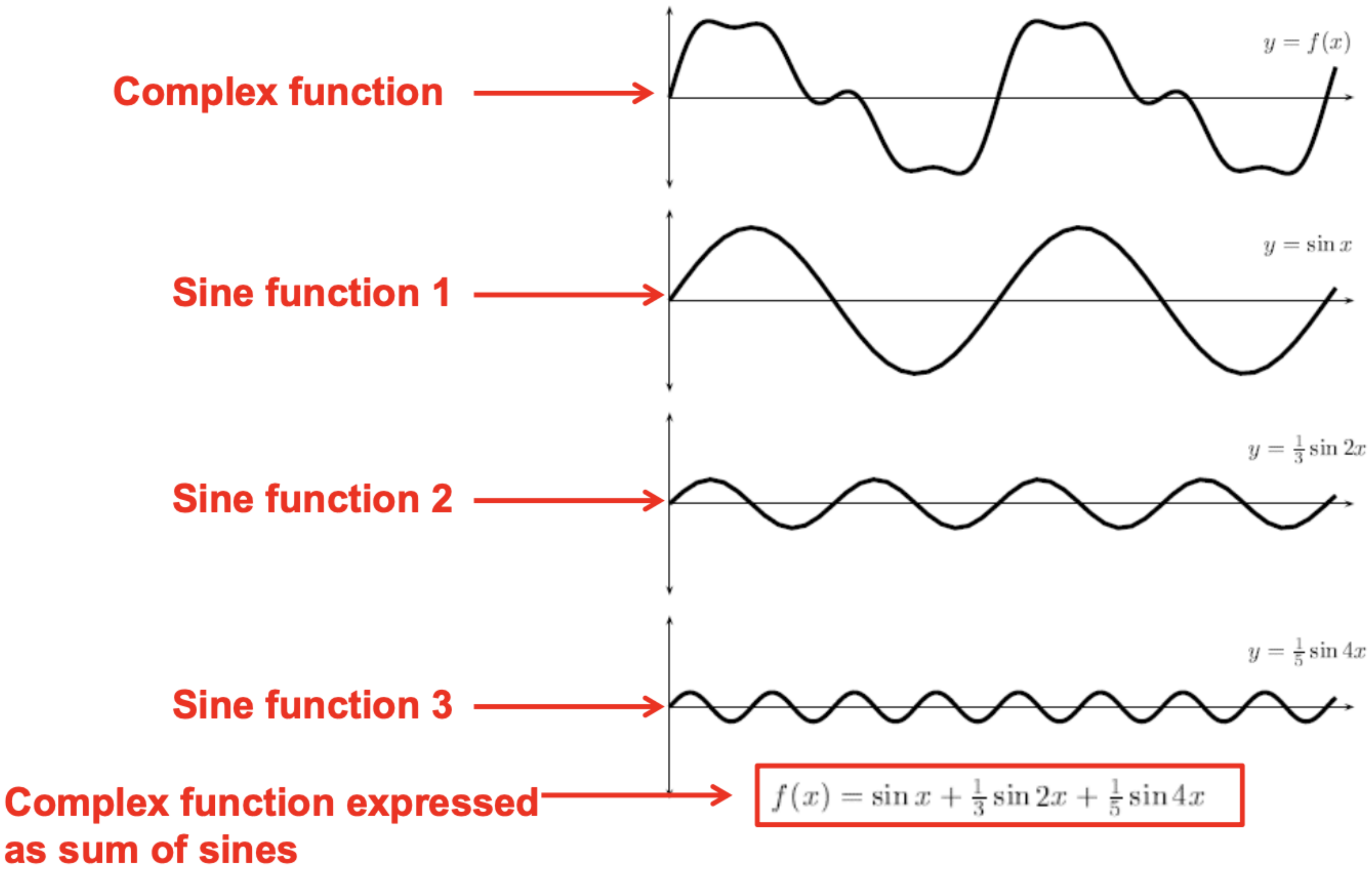
$$y(n)=\sum_{k=0}^{N}h(k)x(n-k) \qquad y(n)=x^{2}(n)+bx(n-k)$$

The non-causal system:

$$y(n)=nx(n+1)+bx^{3}(n-1) \qquad y(n)=\sum_{k=-10}^{10}h(k)x(n-k)$$

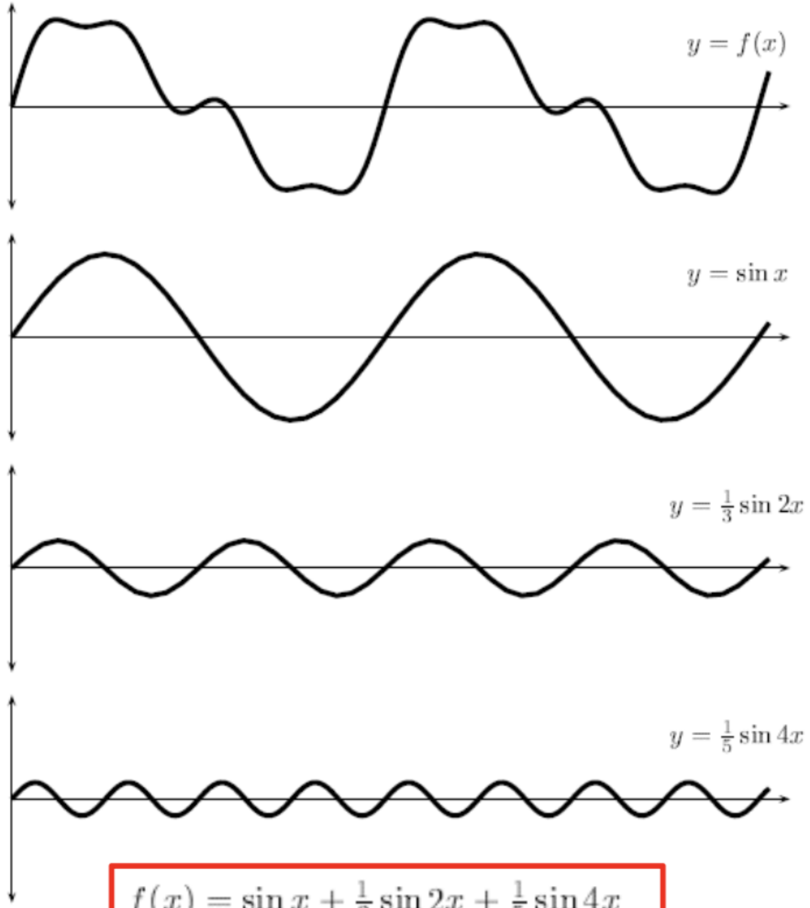
Discrete Fourier Transform

- **Main idea:** Any periodic function can be decomposed into a summation of sines and cosines

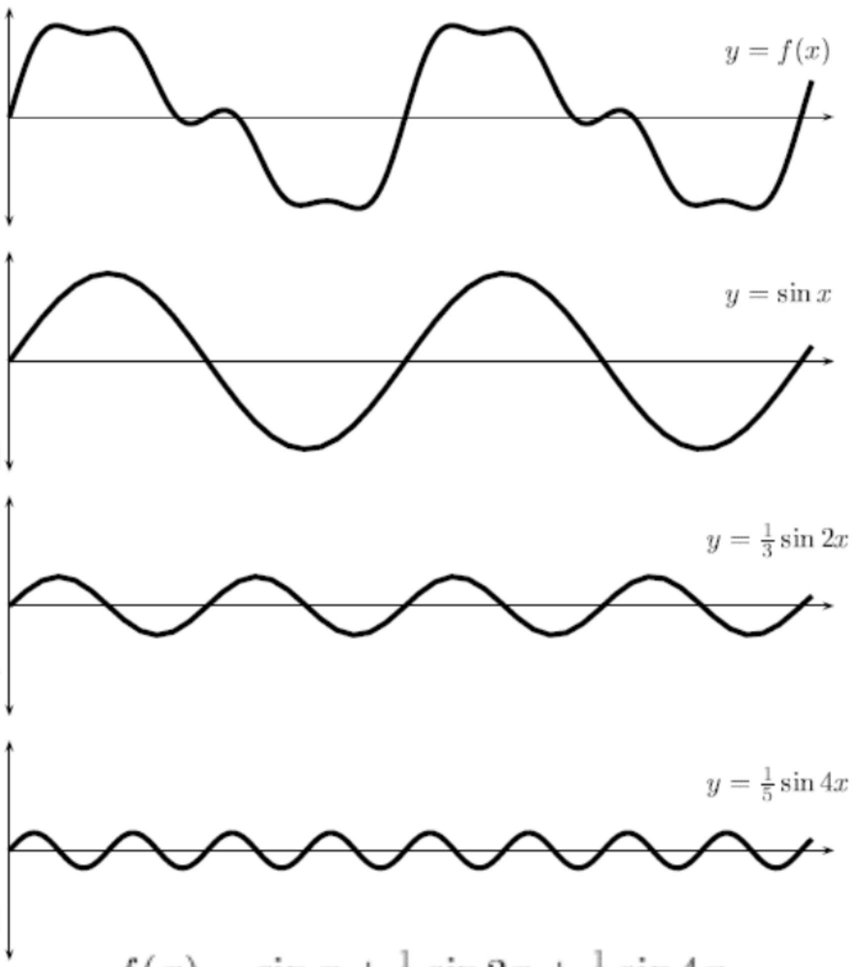


Fourier Transform: Why?

- Mathematatially easier to analyze effects of transmission medium, noise, etc on simple sine functions, then add to get effect on complex signal



$$\boxed{f(x) = \sin x + \frac{1}{3}\sin 2x + \frac{1}{5}\sin 4x}$$



$$f(x) = \sin x + \frac{1}{3}\sin 2x + \frac{1}{5}\sin 4x$$

Observation 2: Frequencies of sines are multiples of each other (called harmonics)

Frequency = 1x

Frequency = 2x

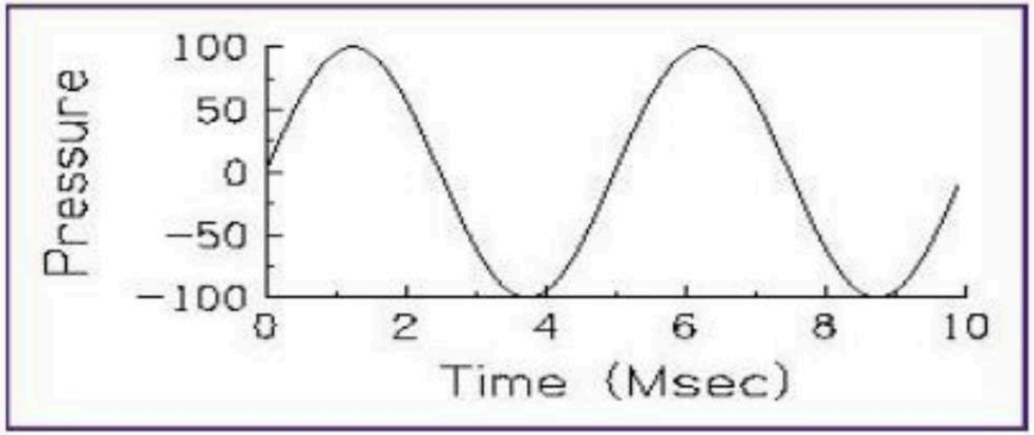
Frequency = 4x

Observation 1: The sines have different frequencies (not same)

Observation 3: Different amounts of different sines added together (e.g. 1/3, 1/5, etc)

Time Domain

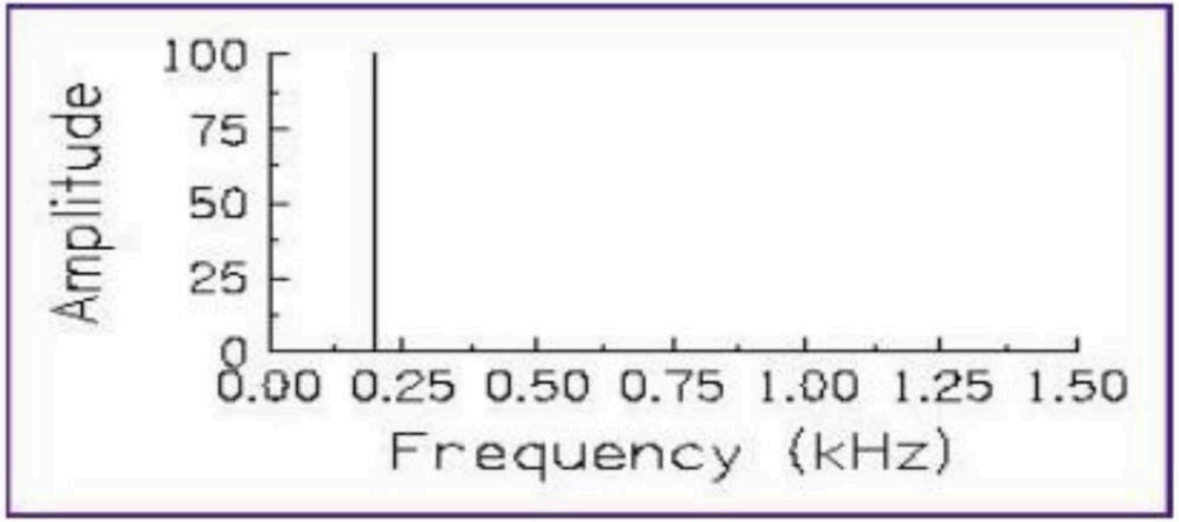
- **Time Domain:**
  - Tells us how properties (air pressure in a sound function, for example) change over time:



- Amplitude = 100
- Frequency = number of cycles in one second = 200 Hz

Frequency Domain

- Frequency domain:
  - Tells us how properties (amplitudes) change over frequencies:



Discrete Fourier Transform

- In practice, we often deal with discrete functions (digital signals, for example)
- Discrete version of the Fourier Transform is much more useful in computer science:

$$f_j = \sum_{k=0}^{n-1} x_k e^{-\frac{2\pi i}{n} jk} \qquad j = 0, \ldots, n-1$$

- $O(n^2)$ time complexity

Find Discrete Fourier Transform (DFT) of x(n) = [2 3 4 4]
Solution:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}nk} \quad for \quad k = 0, 1.., N-1$$

$$e^{-j\frac{\pi}{2}} = cos\frac{\pi}{2} - jsin\frac{\pi}{2} = -j \quad e^{-j\pi} = cos(\pi) - jsin(\pi) = -1$$
$$e^{-j\frac{3\pi}{2}} = cos\frac{3\pi}{2} - jsin\frac{3\pi}{2} = j \quad e^{-j2\pi} = cos(2\pi) - jsin2(\pi) = 1$$

for k=0,1,2,3

$$X(0) = \sum_{n=0}^{3} x(n)e^{0} = \left[2e^{0} + 3e^{0} + 4e^{0} + 4e^{0}\right] = [2 + 3 + 4 + 4] = 13$$

$$X(1) = \sum_{n=0}^{3} x(n)e^{-j\frac{2\pi n}{4}} = \left[2e^{0} + 3e^{-j\pi/2} + 4e^{-j\pi} + 4e^{-j3\pi/2}\right] = [2 - 3j - 4 + 4j] = [-2 + j]$$

$$X(2) = \sum_{n=0}^{3} x(n)e^{\frac{-j4\pi n}{4}} = \left[2e^{0} + 3e^{-j\pi} + 4e^{-j2\pi} + 4e^{-j3\pi}\right] = [2 - 3 + 4 - 4] = [-1 - 0j] = -1$$

$$X(3) = \sum_{n=0}^{3} x(n)e^{\frac{-j6\pi n}{4}} = \left[2e^{0} + 3e^{-j3\pi/2} + 4e^{-j3\pi} + 4e^{-j9\pi/2}\right] = [2 + 3j - 4 - 4j][-2 - j]$$

The DFT of the sequence x(n) = [2 3 4 4] is [13, -2+j, -1, -2-j]

Problem

- Find the DFT of a sequence $x(n) = \{1,1,0,0\}$.

Problem

- Find the DFT of a sequence x(n)= {1,1,0,0}.

{2, 1-j, 0, 0.5}

Convolution

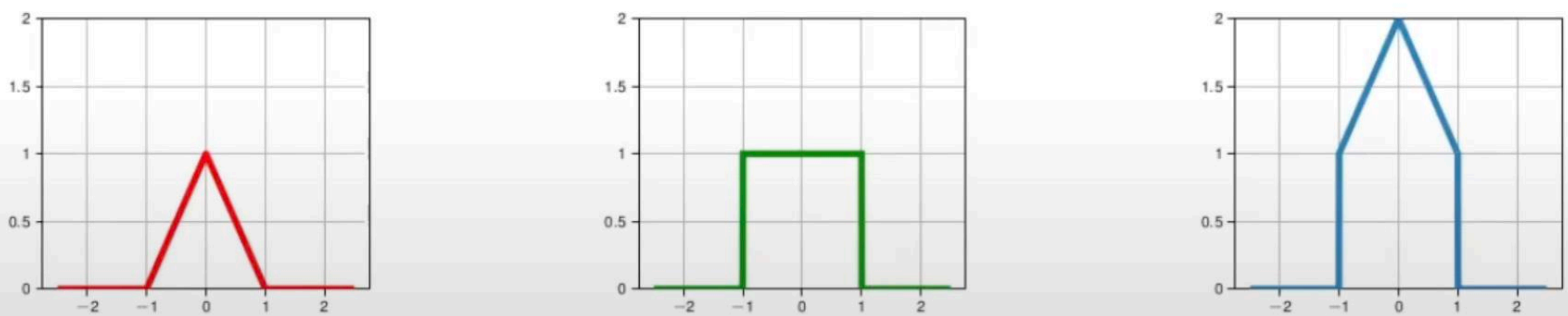
Convolution

- Two functions can be combined several different ways:
- Example of building the sum of two functions:

$$f(t) = \mathrm{rect}(0.5 \cdot t) \cdot (1 - |t|)$$

$$h(t) = \mathrm{rect}(0.5 \cdot t)$$

$$g(t) = f(t) + h(t)$$



$f(t)$ $h(t)$ $f(t) + h(t)$

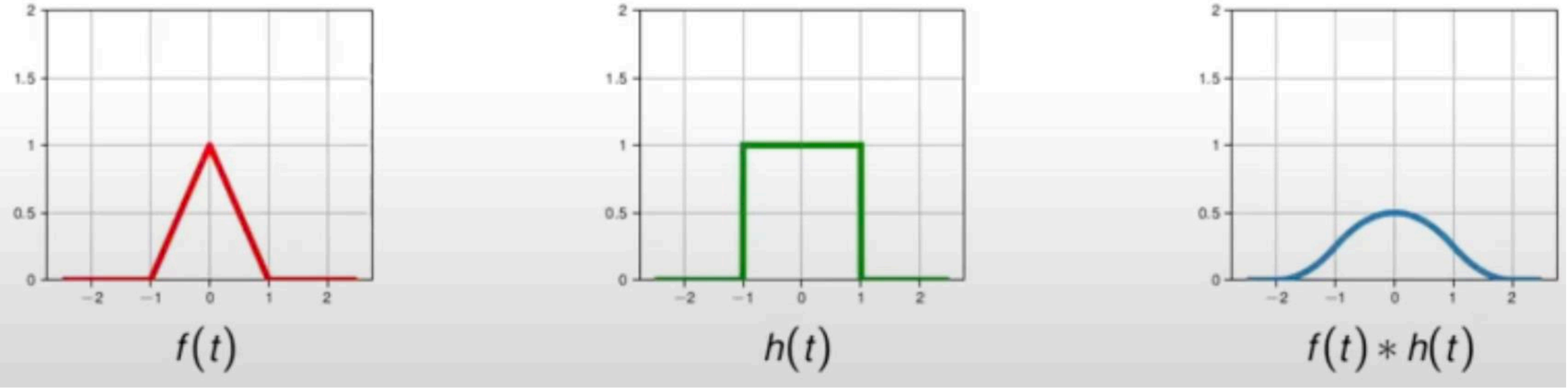
Convolution

- Convolution is just a different way of combining two functions:

$$f(t) = \mathrm{rect}(0.5 \cdot t) \cdot (1 - |t|)$$

$$h(t) = \mathrm{rect}(0.5 \cdot t)$$

$$g(t) = f(t) * h(t)$$



Convolution

- Convolution is an integral that expresses the amount of overlap of one function when it is shifted over another function.

- How the first function modifies the second function.

1D Continuous Convolution

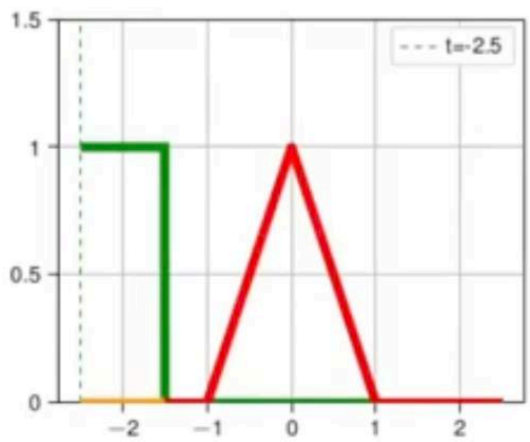
- Convolution is defined as follows:

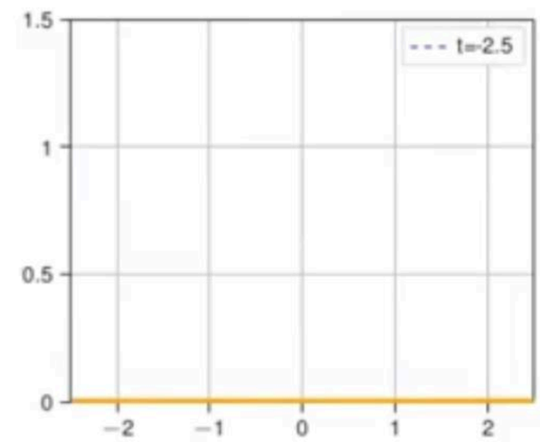
$$f(x) * g(x) = \int_{-\infty}^{\infty} f(a)g(x-a)da$$

Impulse Response

- The IRF is a mathematical representation of how a system responds to an impulse input.
- An impulse input is a brief burst of energy that has an infinite amplitude and an infinitesimal duration.
- When we apply this input to a system, it responds in a particular way, and this response can be represented by the impulse response function.
- The impulse response is defined as the output of a system when a unit impulse is applied to the input of the system.
- It is a time-domain representation that describes how the system responds to sudden changes in the input.
- The IRF provides valuable insight into how the system processes and modifies input signals.

How can convolution be visualized ?





$$f(\tau) = \mathrm{rect}\,(0.5 \cdot \tau) \cdot (1 - |\tau|)$$

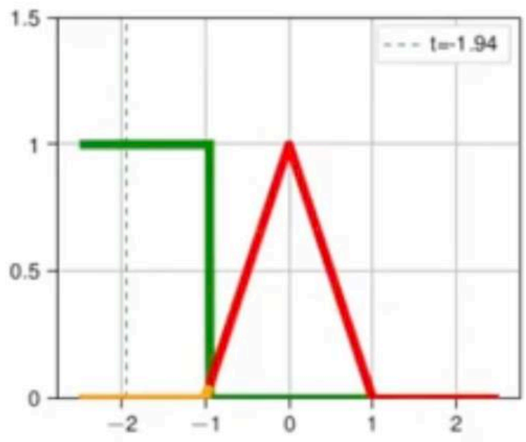
$$h(t - \tau) = \mathrm{rect}\,(0.5 \cdot (t - \tau))$$

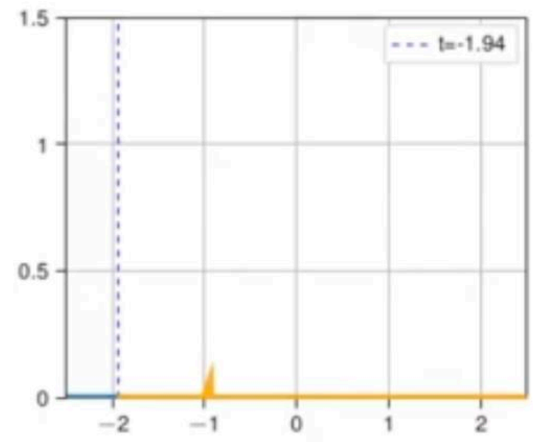
$$f(\tau) \cdot h(t - \tau)$$

$$g(t) = f(t) * h(t)$$

$$= \int_{-\infty}^{+\infty} f(\tau) \cdot h(t - \tau)\, \mathrm{d}\,\tau$$

How can convolution be visualized ?





$$f(\tau) = \mathrm{rect}(0.5\cdot\tau)\cdot(1-|\tau|)$$

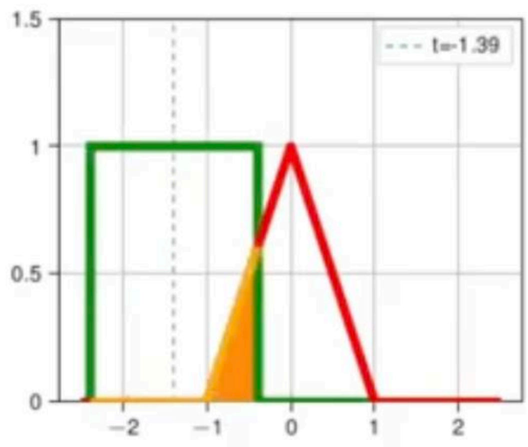
$$h(t-\tau) = \mathrm{rect}(0.5\cdot(t-\tau))$$

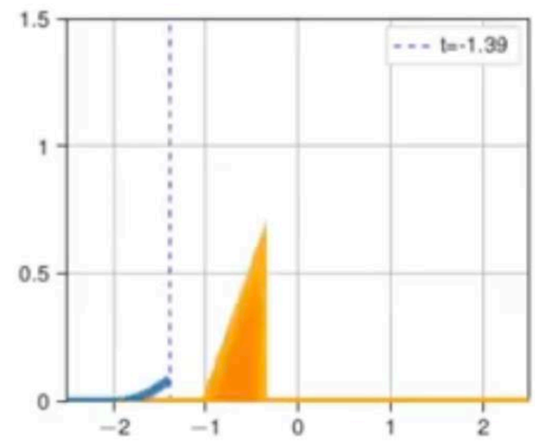
$$f(\tau)\cdot h(t-\tau)$$

$$g(t) = f(t) * h(t)$$

$$= \int_{-\infty}^{+\infty} f(\tau)\cdot h(t-\tau)\,\mathrm{d}\tau$$

How can convolution be visualized ?





$$f(\tau) = \mathrm{rect}(0.5 \cdot \tau) \cdot (1 - |\tau|)$$

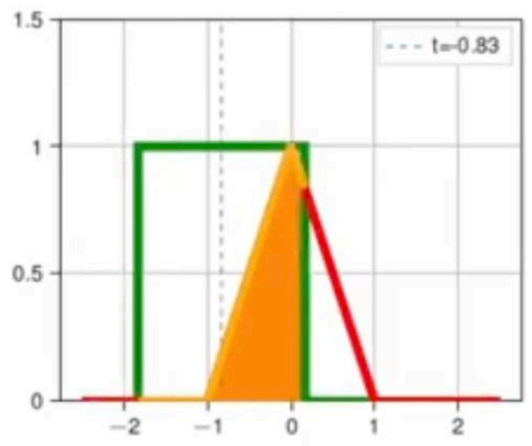
$$h(t-\tau) = \mathrm{rect}(0.5 \cdot (t-\tau))$$

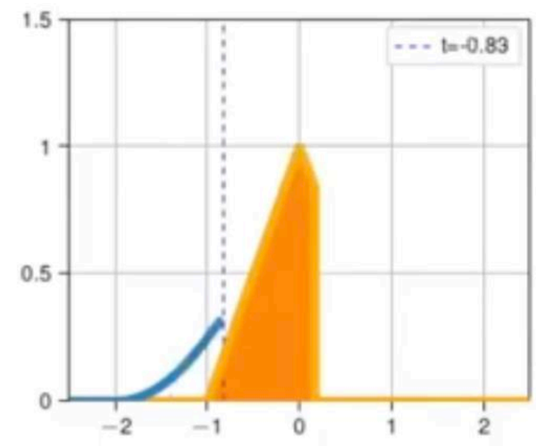
$$f(\tau) \cdot h(t-\tau)$$

$$g(t) = f(t) * h(t)$$

$$= \int_{-\infty}^{+\infty} f(\tau) \cdot h(t-\tau) \, d\tau$$

How can convolution be visualized ?





$$f(\tau) = \mathrm{rect}(0.5 \cdot \tau) \cdot (1 - |\tau|)$$

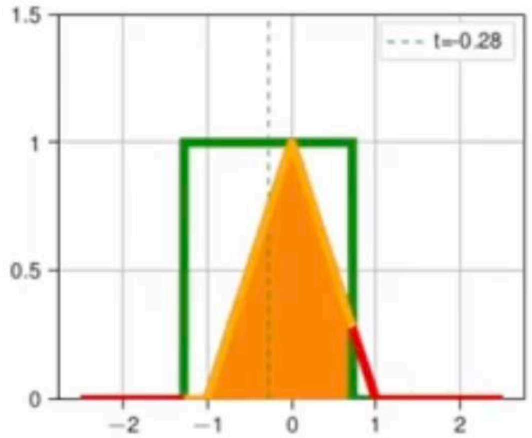
$$h(t - \tau) = \mathrm{rect}(0.5 \cdot (t - \tau))$$

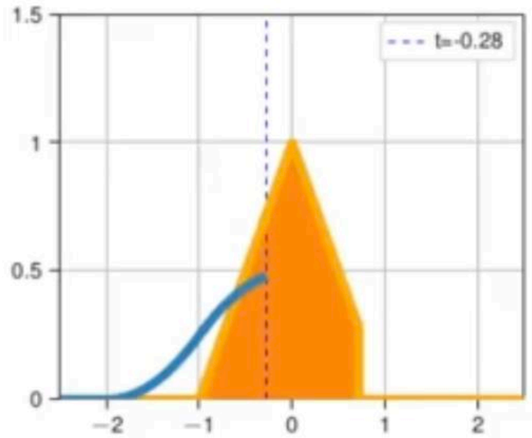
$$f(\tau) \cdot h(t - \tau)$$

$$g(t) = f(t) * h(t)$$

$$= \int_{-\infty}^{+\infty} f(\tau) \cdot h(t - \tau) \, d\tau$$

How can convolution be visualized ?





$$f(\tau) = \mathrm{rect}\,(0.5 \cdot \tau) \cdot (1 - |\tau|)$$

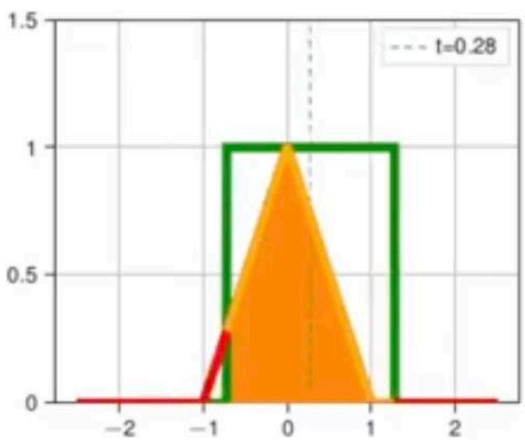
$$h(t-\tau) = \mathrm{rect}\,(0.5 \cdot (t-\tau))$$

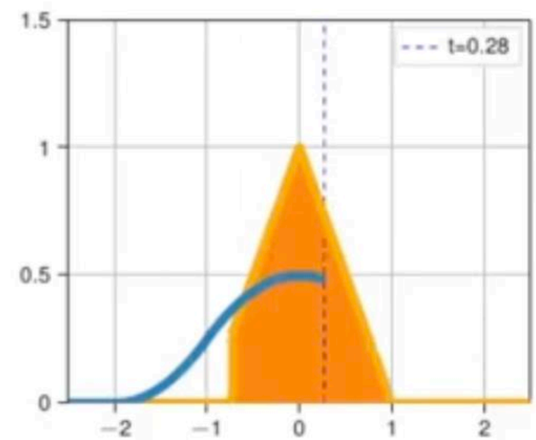
$$f(\tau) \cdot h(t-\tau)$$

$$g(t) = f(t) * h(t)$$

$$= \int_{-\infty}^{+\infty} f(\tau) \cdot h(t-\tau)\,\mathrm{d}\,\tau$$

How can convolution be visualized ?





$$f(\tau) = \mathrm{rect}(0.5 \cdot \tau) \cdot (1 - |\tau|)$$

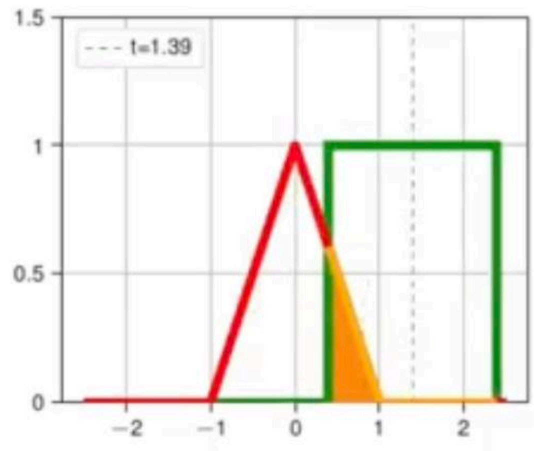
$$h(t-\tau) = \mathrm{rect}(0.5 \cdot (t-\tau))$$

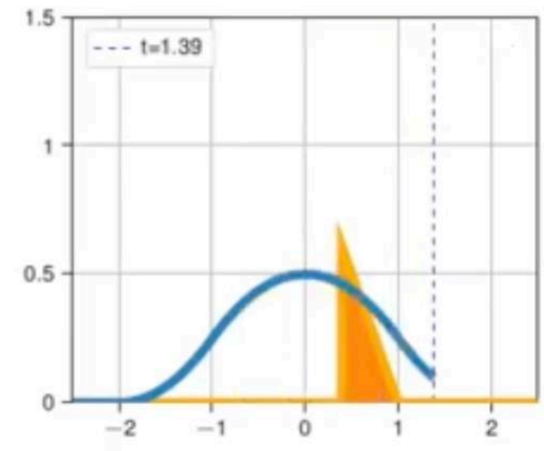
$$f(\tau) \cdot h(t-\tau)$$

$$g(t) = f(t) * h(t)$$

$$= \int_{-\infty}^{+\infty} f(\tau) \cdot h(t-\tau)\, \mathrm{d}\tau$$

How can convolution be visualized ?





$$f(\tau) = \mathrm{rect}\,(0.5\cdot\tau)\cdot(1-|\tau|)$$

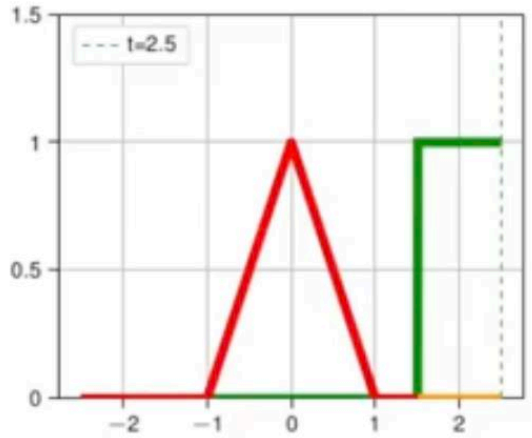
$$h(t-\tau) = \mathrm{rect}\,(0.5\cdot(t-\tau))$$

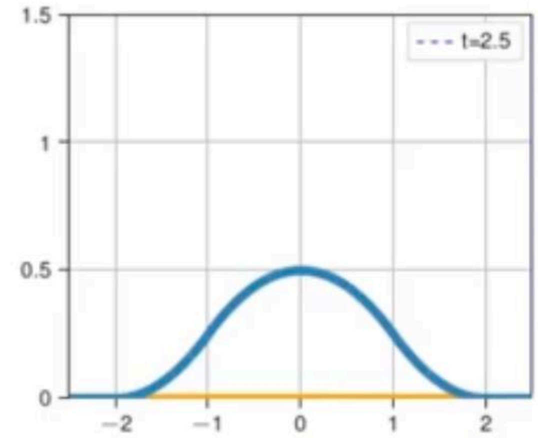
$$f(\tau)\cdot h(t-\tau)$$

$$g(t) = f(t) * h(t)$$

$$= \int_{-\infty}^{+\infty} f(\tau)\cdot h(t-\tau)\,\mathrm{d}\,\tau$$

How can convolution be visualized ?





$$f(\tau) = \mathrm{rect}\,(0.5 \cdot \tau) \cdot (1 - |\tau|)$$

$$h(t - \tau) = \mathrm{rect}\,(0.5 \cdot (t - \tau))$$

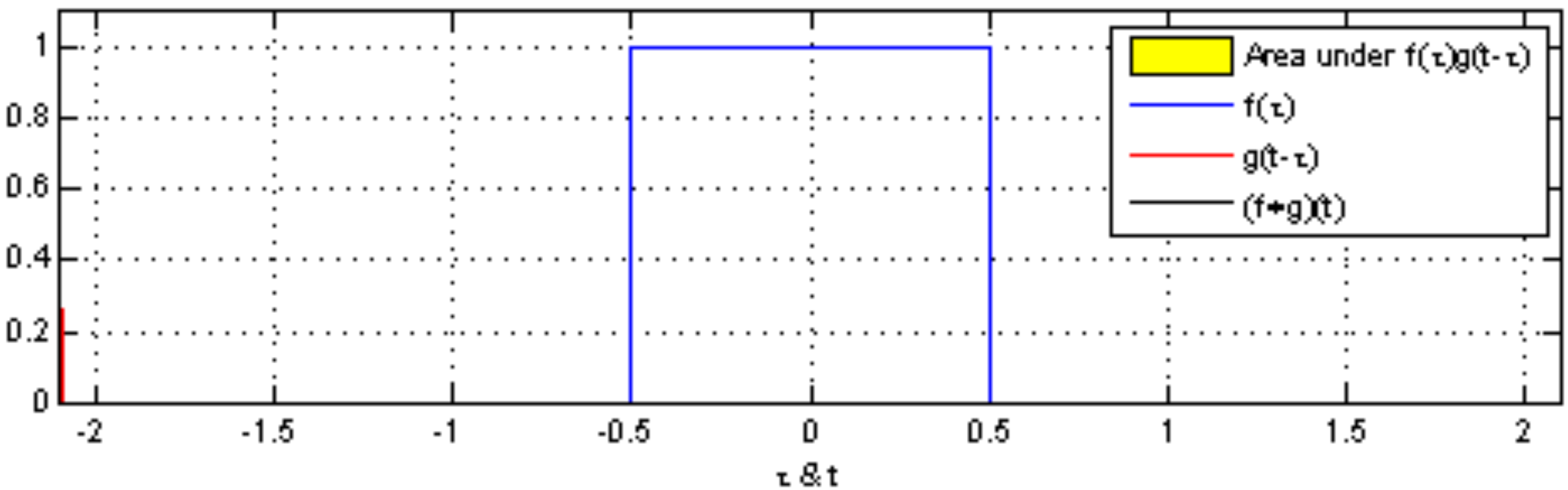
$$f(\tau) \cdot h(t - \tau)$$

$$g(t) = f(t) * h(t)$$

$$= \int_{-\infty}^{+\infty} f(\tau) \cdot h(t - \tau)\,\mathrm{d}\,\tau$$

How can convolution be visualized ?

$$f(x) * g(x) = \int_{-\infty}^{\infty} f(a)g(x-a)da$$



2D Convolution

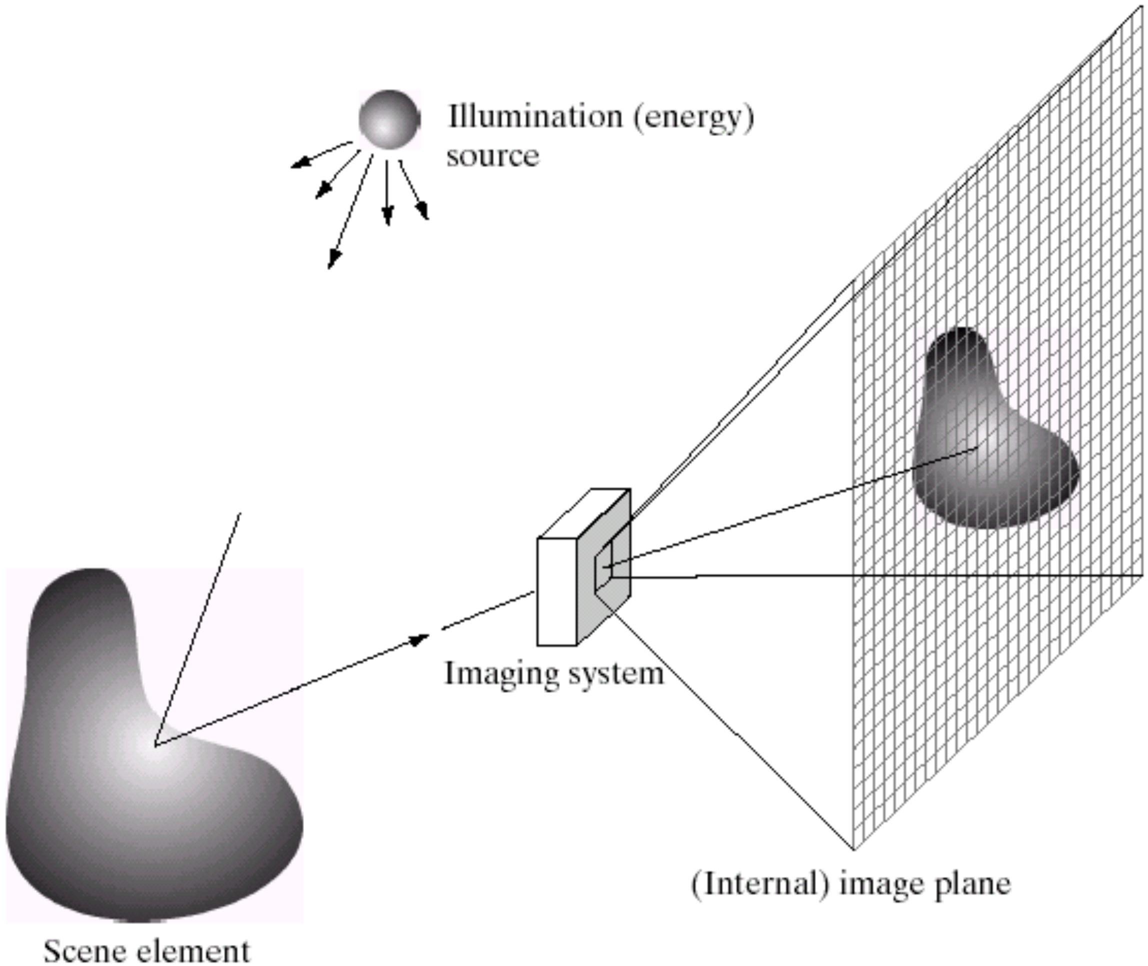
2D Signals

- An image is defined as a two-dimensional function, **F(x,y)**, where x and y are spatial coordinates, and the amplitude of **F** at any pair of coordinates (x,y) is called the **intensity** of that image at that point.

2D Signals

- Digital camera samples light to obtain x(m, n).
- Color Images - > 3 values at each pixel (R, G, B)

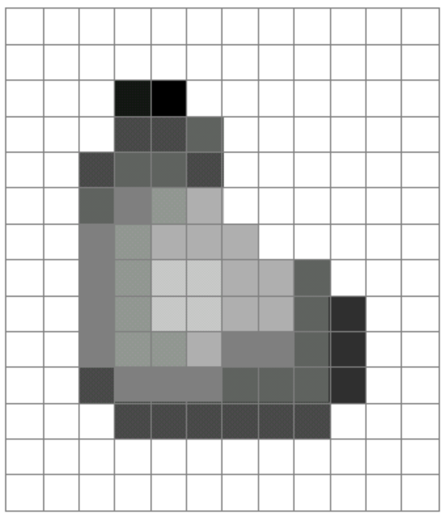
2D Signals



Illumination (energy)
source
Imaging system
(Internal) image plane
Scene element

2D Signals

- A grid (matrix) of intensity values



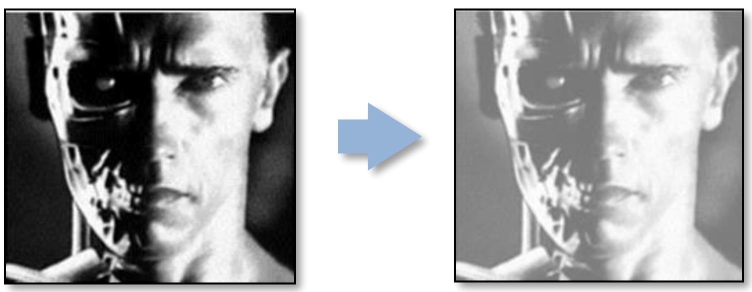
=

255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	20	0	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255
255	255	127	145	175	175	175	255	255	255	255	255
255	255	127	145	200	200	175	175	95	255	255	255
255	255	127	145	200	200	175	175	95	47	255	255
255	255	127	145	145	175	127	127	95	47	255	255
255	255	74	127	127	127	95	95	95	47	255	255
255	255	255	74	74	74	74	74	74	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255

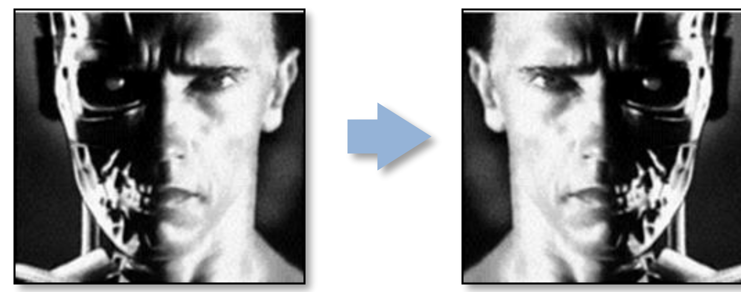
- (common to use one byte per value: 0 = black, 255 = white)

Image Transformations

- As with any function, we can apply operators to an image



$g(x,y) = f(x,y) + 20$



$g(x,y) = f(-x,y)$

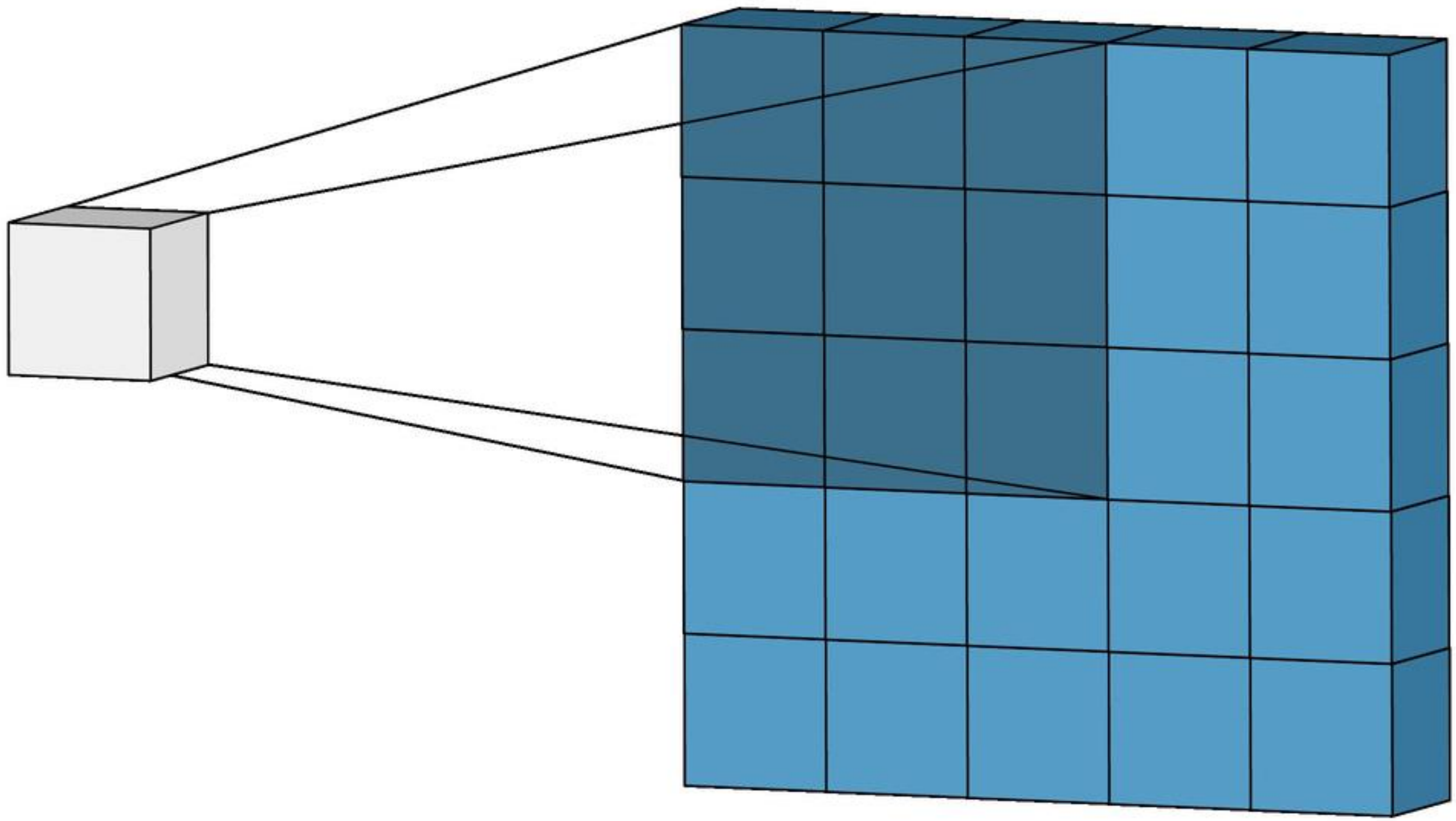
2D Convolution

2D convolution operation is defined as:

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u,j-v]$$

$$G = H * F$$

- Convolution is **commutative** and **associative**



Convolution

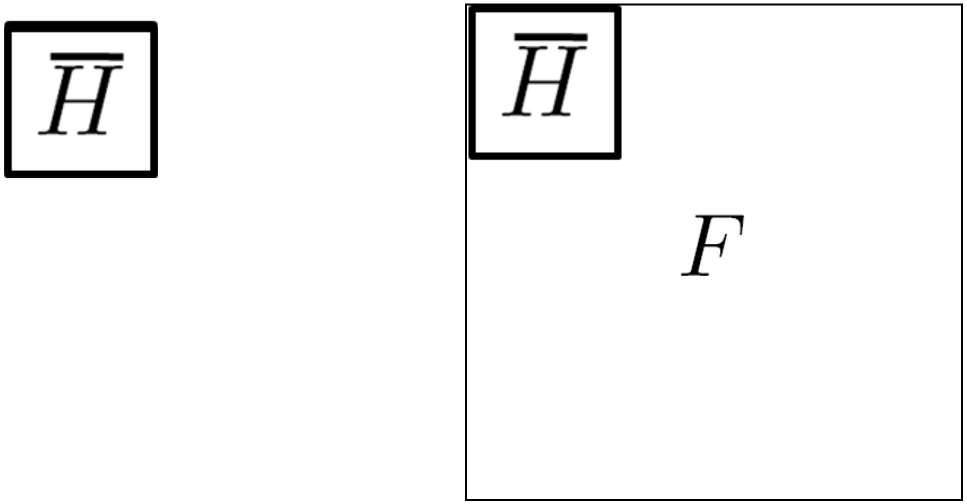


Image Filtering

- An FIR filter computes the output as the weighted sum of different values.

$$y[n] = (1/3)x[n] + (1/3)x[n-1] + (1/3)x[n-2]$$

- 2D filtering is similar.

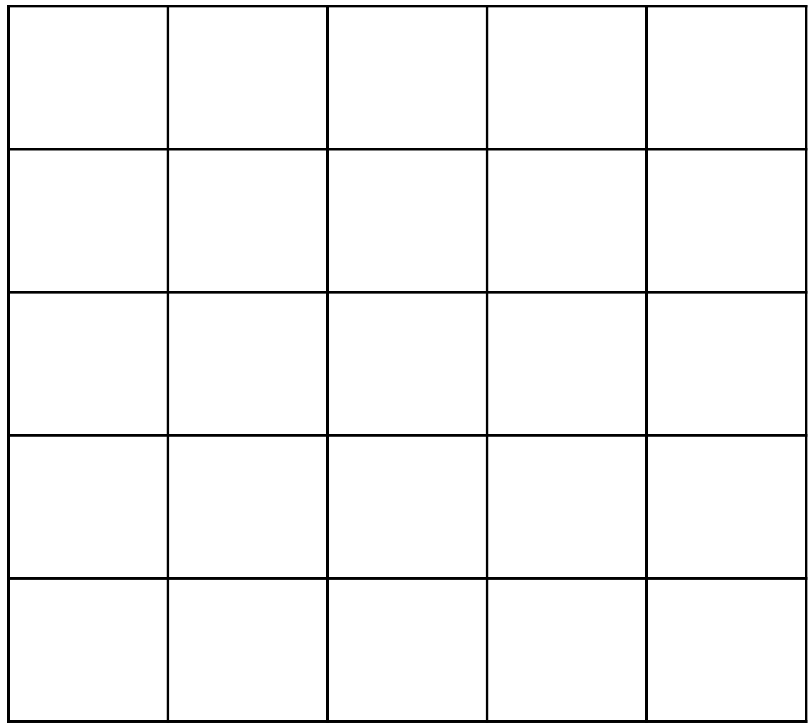
$$y[m, n] = (1/3)x[m, n] + (1/3)x[m, n] + (1/3)x[m, n]$$

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u, j-v]$$

Convolution Operation

Filter / Kernel (3 x 3)

Convolution Operation



Filter / Kernel (5 x 5)

An image kernel is a small matrix that is used to apply the effects like the ones you may find in Photoshop such as blurring, sharpening, outlining or embossing.

Convolution Operation

20	**24**	**11**	**12**	**16**	**19**
19	**17**	**20**	**23**	**15**	**9**
21	**40**	**25**	**13**	**14**	**8**
9	**18**	**8**	**6**	**11**	**22**
31	**3**	**7**	**9**	**17**	**23**
20	**12**	**3**	**11**	**19**	**30**

Input Slice (6x6)

*

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

Kernel (3x3)

=

Convolution Operation

20	**24**	**11**	**12**	**16**	**19**
19	**17**	**20**	**23**	**15**	**9**
21	**40**	**25**	**13**	**14**	**8**
9	**18**	**8**	**6**	**11**	**22**
31	**3**	**7**	**9**	**17**	**23**
20	**12**	**3**	**11**	**19**	**30**

Input Slice (6x6)

$$*\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} =$$

Sobel Kernel

Convolution Operation

Receptive Field

20	**24**	**11**	**12**	**16**	**19**
19	**17**	**20**	**23**	**15**	**9**
21	**40**	**25**	**13**	**14**	**8**
9	**18**	**8**	**6**	**11**	**22**
31	**3**	**7**	**9**	**17**	**23**
20	**12**	**3**	**11**	**19**	**30**

Input Slice (6x6)

*

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

Sobel Kernel

=

Convolution Operation

$$(20 \times 1) + (24 \times 0) + (11 \times -1) + (19 \times 2) + (17 \times 0) + (20 \times -2) + (21 \times 1) + (40 \times 0) + (25 \times -1) = -3$$

20	24	11	12	16	19
19	17	20	23	15	9
21	40	25	13	14	8
9	18	8	6	11	22
31	3	7	9	17	23
20	12	3	11	19	30

Input Slice (6x6)

*

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

Sobel Kernel (3x3)

=

-3			

Output (4x4)

Convolution Operation

Stride of one

20	24	11	12	16	19
19	17	20	23	15	9
21	40	25	13	14	8
9	18	8	6	11	22
31	3	7	9	17	23
20	12	3	11	19	30

Input Slice (6x6)

*

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

Sobel Kernel (3x3)

=

-3	27		

Output (4x4)

(20 x 1) +(24 x 0) +(11 x -1)
+(19 x 2) +(17 x 0) +(20 x -2)
+(21 x 1) +(40 x 0) + (25 x -1) = -3

(24 x 1) +(11 x 0) +(12 x -1)
+(17 x 2) +(20 x 0) +(23 x -2)
+(40 x 1) +(25 x 0) + (13 x -1) = 27

Convolution Operation

20	**24**	**11**	**12**	**16**	**19**
19	**17**	**20**	**23**	**15**	**9**
21	**40**	**25**	**13**	**14**	**8**
9	**18**	**8**	**6**	**11**	**22**
31	**3**	**7**	**9**	**17**	**23**
20	**12**	**3**	**11**	**19**	**30**

Input Slice (6x6)

*

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

Sobel Kernel (3x3)

=

-3	27		

Output (4x4)

The output from the convolution operation has a smaller spatial size than the input slice.

How the filter is placed above the input data so that it doesn't extent beyond the boundaries of the input data.

Convolution Operation

20	24	11	12	16	19
19	17	20	23	15	9
21	40	25	13	14	8
9	18	8	6	11	22
31	3	7	9	17	23
20	12	3	11	19	30

Input Slice (6x6)

*

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

Sobel Kernel (3x3)

=

-3	27		

Output (4x4)

Padding techniques are used to obtain output with the same size as that of the input.

Convolution Operation

Stride of two

20	24	11	12	16	19
19	17	20	23	15	9
21	40	25	13	14	8
9	18	8	6	11	22
31	3	7	9	17	23
20	12	3	11	19	30

Input Slice (6x6)

*

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

Sobel Kernel (3x3)

=

-3	27

Output (4x4)

Example

Image Filter

- Compute function of local neighborhood at each position – Image Filtering

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

- Really Important!
  - Enhance Image – Denoise, resize, increase contrast, etc
  - Detect pattern – Template matching
  - Extract Information - Texture, edges, distinctive points

Linear Filters

- Linearity

imfilter(I, f1+f2) = imfilter(I, f1) + imfilter(I, f2)

- Shift Invariance

Same behavior regardless of pixel position.

imfilter(I, shift(f1)) = shift(imfilter(I, f1))

Mean Filter (Box Filter)

- Simple, Intuitive and easy to implement method of smoothening images.

- Used to reduce the noise in the image.

- The idea of mean filtering is simply to replace each pixel value in an image with the mean (`average') value of its neighbors, including itself.

Mean filtering

H

$*$

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

F

$=$

	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	

G

Gaussian Filter

- The Gaussian smoothing operator is a 2D convolution operator that is used to 'blur' images and remove noise.

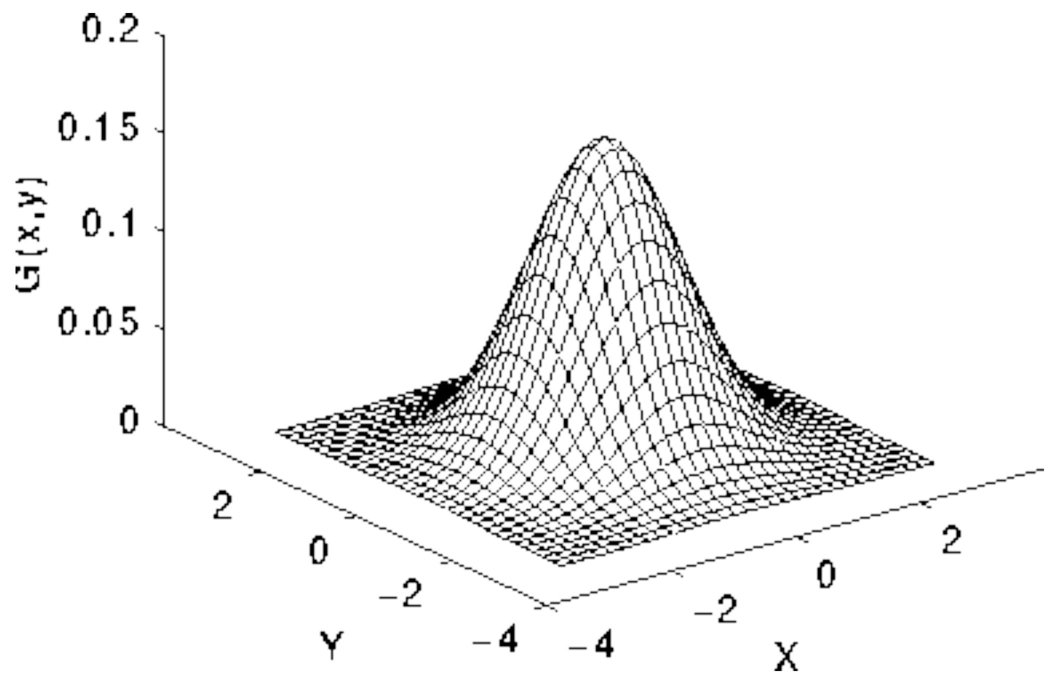
- Similar to mean filter, uses a different kernel that represents the shape of a Gaussian.

- The degree of smoothing is determined by the standard deviation of the Gaussian.

Gaussian Filter

In 2D, Gaussian has the form:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



Gaussian distribution with mean = (0,0) and SD =1

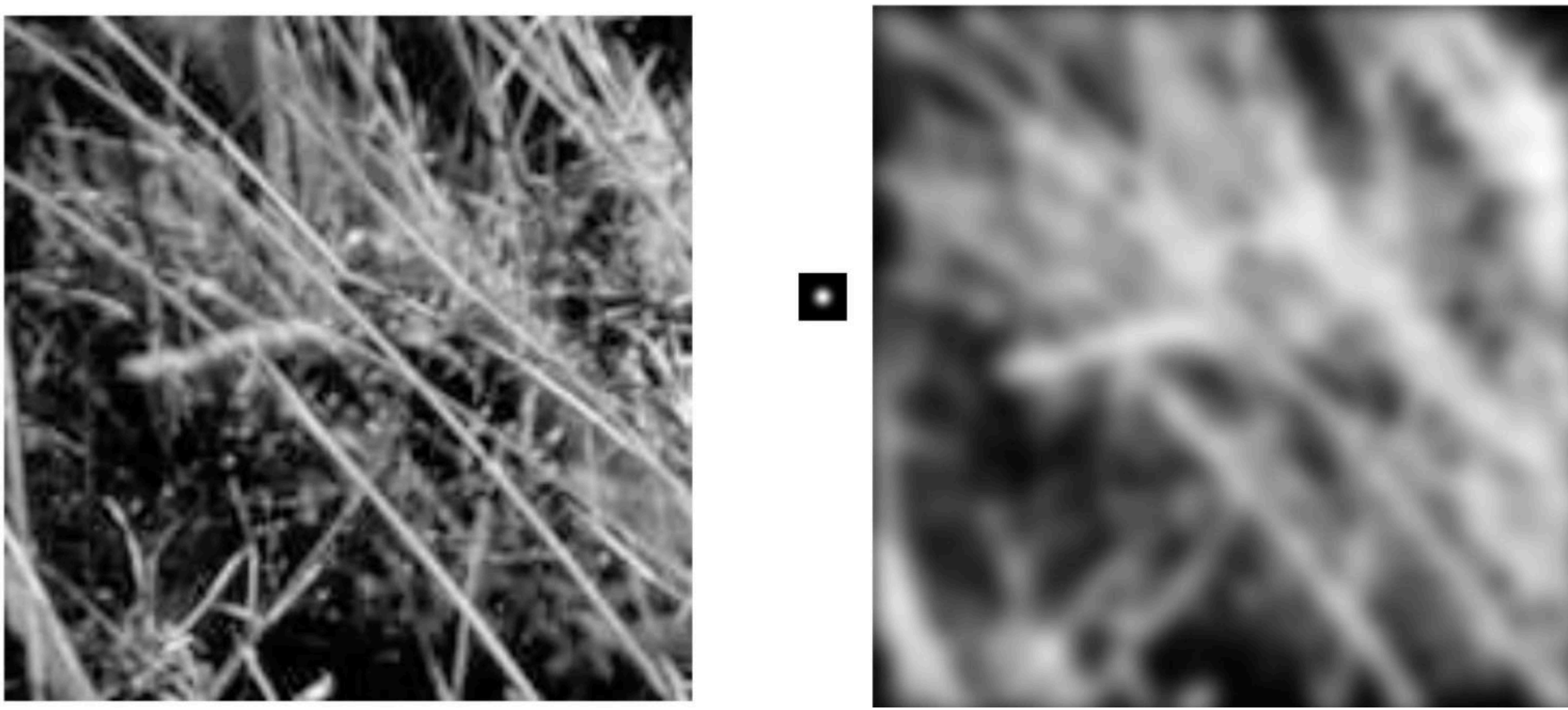
Gaussian Filter

- An integer valued 5 by 5 convolution kernel approximating a Gaussian with a σ of 1 is shown below

$$\frac{1}{273}$$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

Smoothing with Gaussian Filter



Gaussian Filter - Properties

- Low-Pass filter:
  - Removes high frequency components from the image.
  - Image becomes more smooth

- Gaussian is convolved with Gaussian is another Gaussian

Linear filters: examples

$$* \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$

Original

Identical image

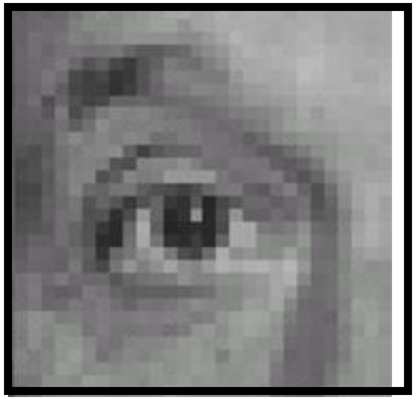
Linear filters: examples

Original

*

0	0	0
1	0	0
0	0	0

=

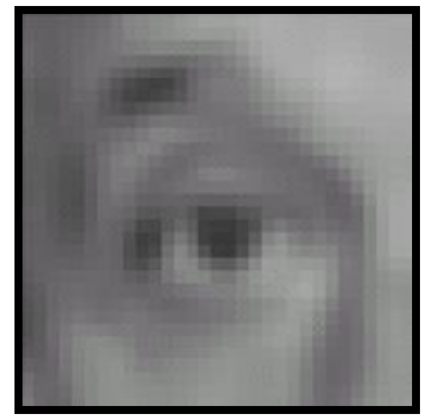


Shifted left
By 1 pixel

Linear filters: examples

Original

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



Blur (with a mean filter)

Source: D. Lowe

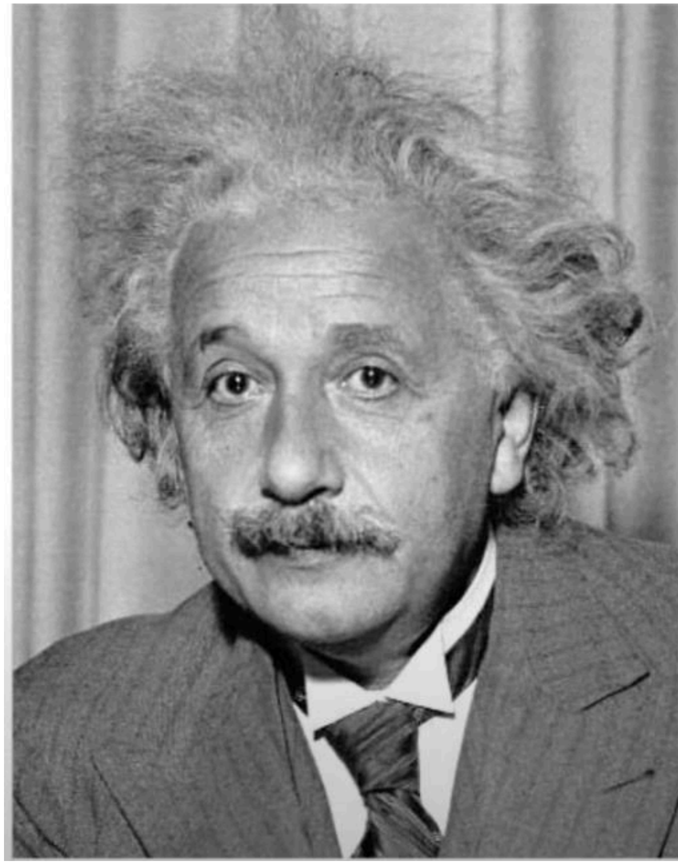
Linear filters: examples

Original

$$* \left(\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} - \frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \right) =$$

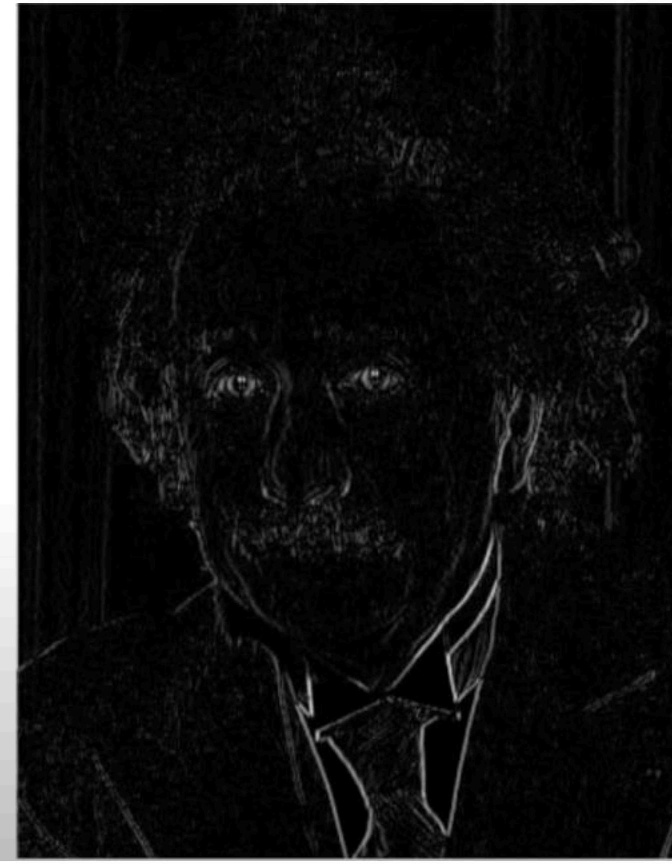
Sharpening filter
(accentuates edges)

Sobel Filter



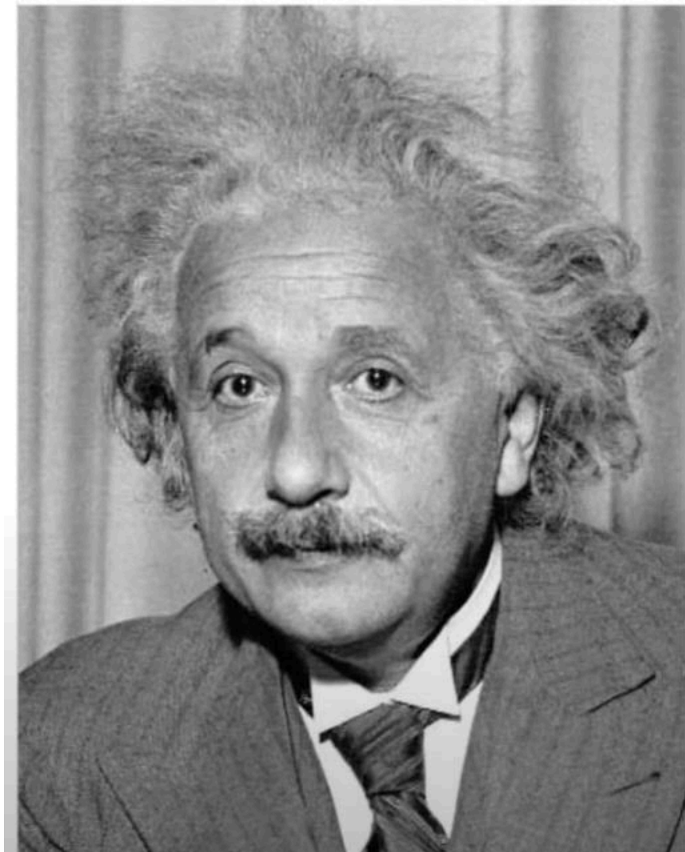
1	0	-1
2	0	-2
1	0	-1

Sobel



Vertical Edge

Sobel Filter



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

Sobel

Horizontal Edge

Sobel Filter

- Edge Enhancing
- Slight Smoothing
- Positive and Negative Values

Image Filters

- Linear Filter - Replace each pixel with a linear combination of its neighbors.

- Nonlinear Filter

Non-Linear Filters

- Min Filter
- Max Filter
- Median Filter

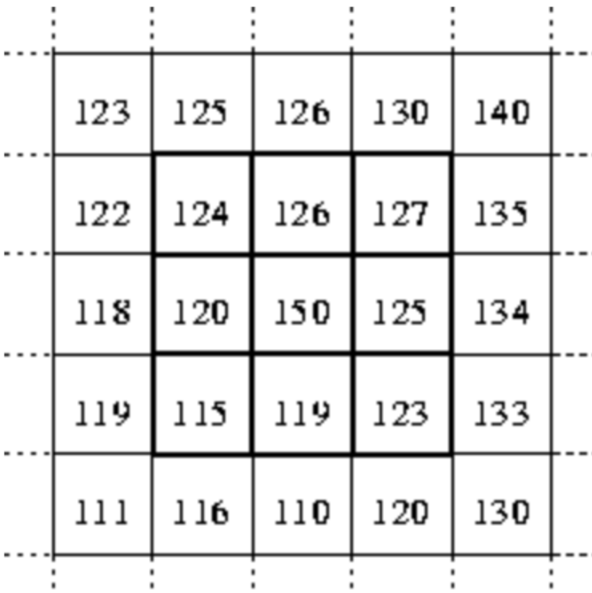
Median Filter

- Considers each pixel in the image in turn and looks at its nearby neighbors to decide whether or not it is representative of its surroundings.
- Replaces with the *median* of those values.

How to calculate the median?

- Sort all the pixel values from the surrounding neighborhood into numerical order.
- Replace the pixel being considered with the middle pixel value.

Median Filter



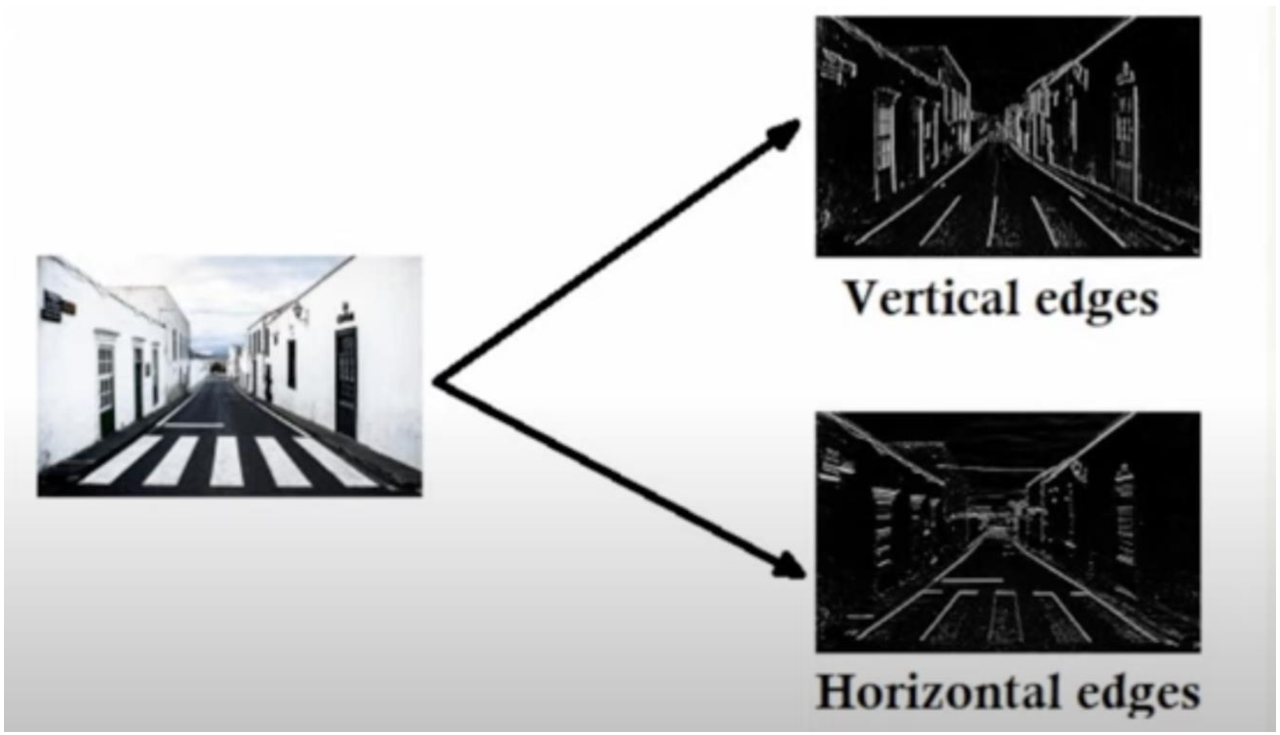
Neighbourhood values:

115, 119, 120, 123, 124,
125, 126, 127, 150

Median value: 124

Edge Filter

- Abrupt change in intensity values.
- Example: Sobel Filter



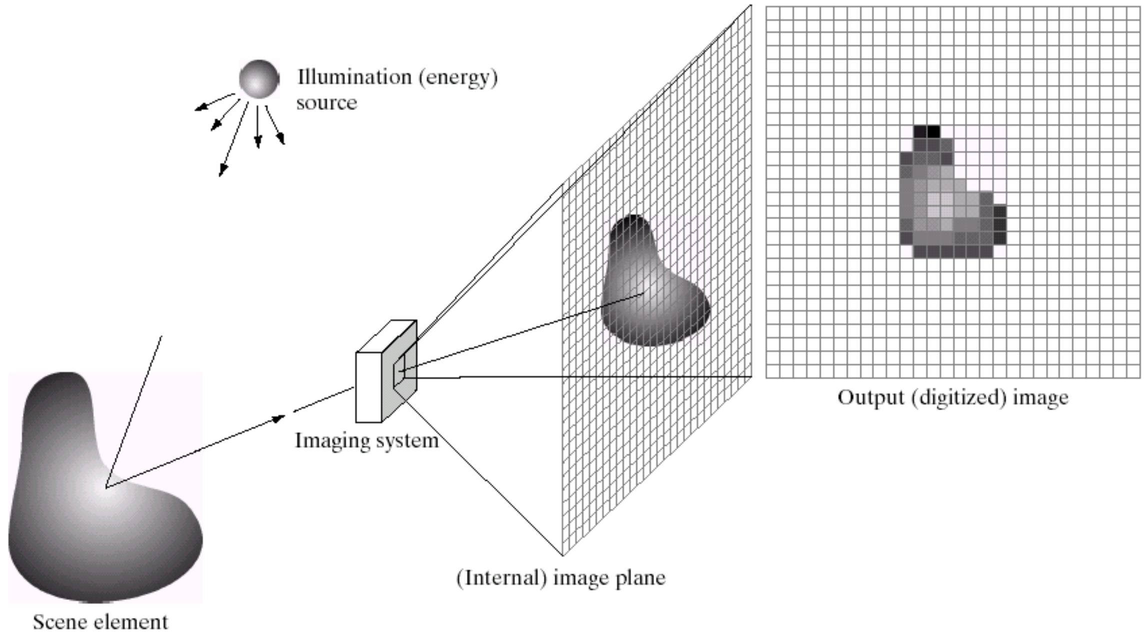
Lecture 4 - Digital Image Processing

Key Take Aways

- What is a digital image?
- What is digital image processing?
- Key stages in digital image processing

What is a Digital Image?

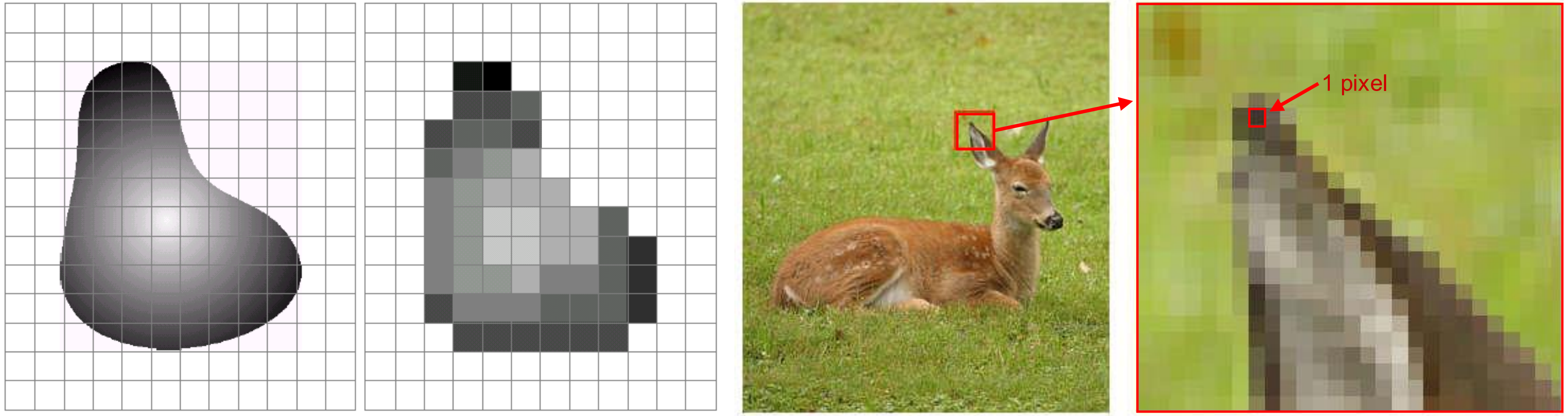
- A **digital image** is a representation of a two-dimensional image as a finite set of digital values, called picture elements or pixels.



What is a Digital Image?

Pixel values typically represent gray levels, colours, heights, opacities etc

Remember *digitization* implies that a digital image is an *approximation* of a real scene



What is a Digital Image?

Common image formats include:

- 1 sample per point (B&W or Grayscale)
- 3 samples per point (Red, Green, and Blue)
- 4 samples per point (Red, Green, Blue, and "Alpha", a.k.a. Opacity)



What is Digital Image Processing?

Digital image processing focuses on two major tasks

- Improvement of pictorial information for human interpretation
- Processing of image data for storage, transmission and representation for autonomous machine perception

Some argument about where image processing ends and fields such as image analysis and computer vision start

What is Digital Image Processing?

Low Level Process

Input: Image
Output: Image

Examples: Noise removal, image sharpening

Mid Level Process

Input: Image
Output: Attributes

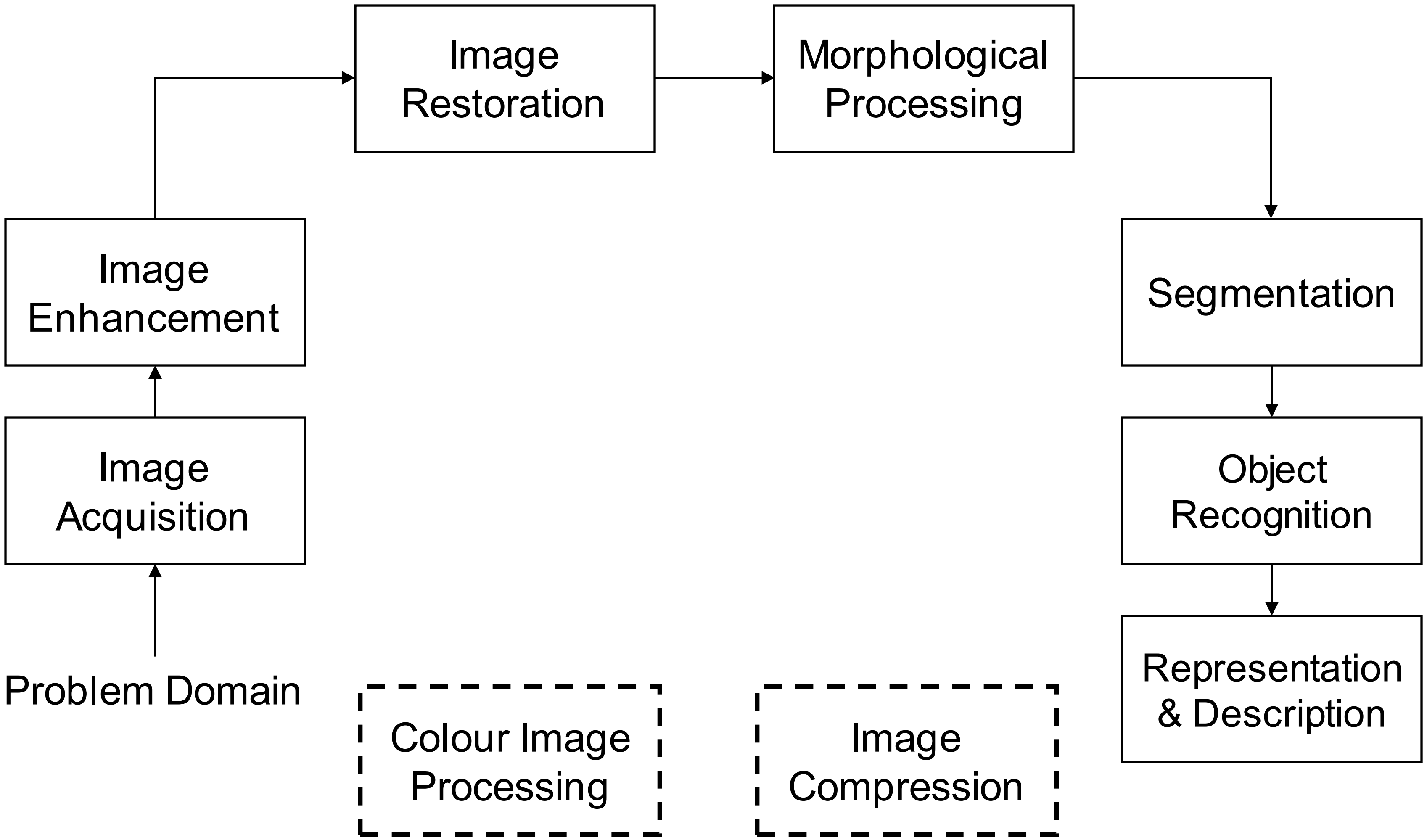
Examples: Object recognition, segmentation

High Level Process

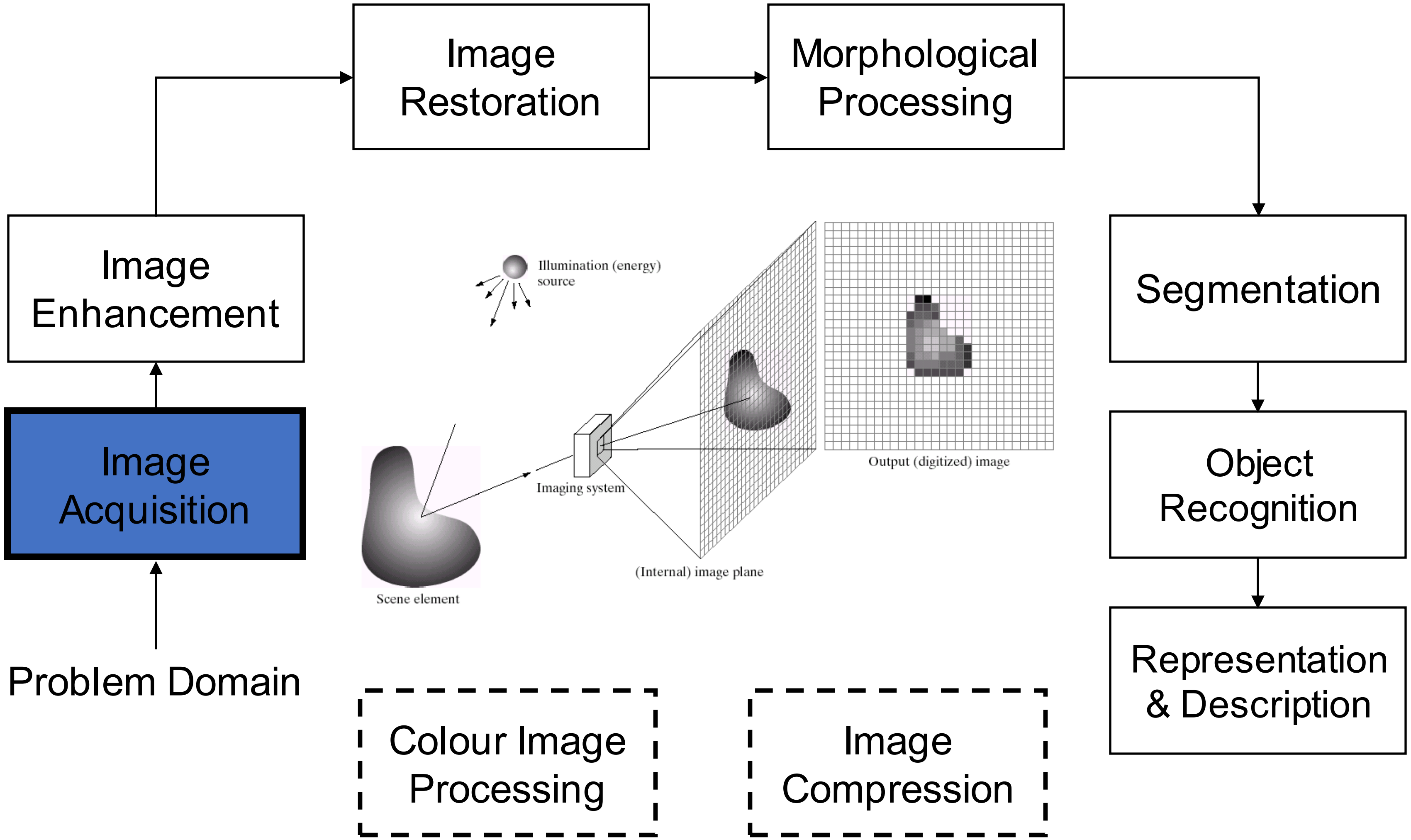
Input: Attributes
Output: Understanding

Examples: Scene understanding, autonomous navigation

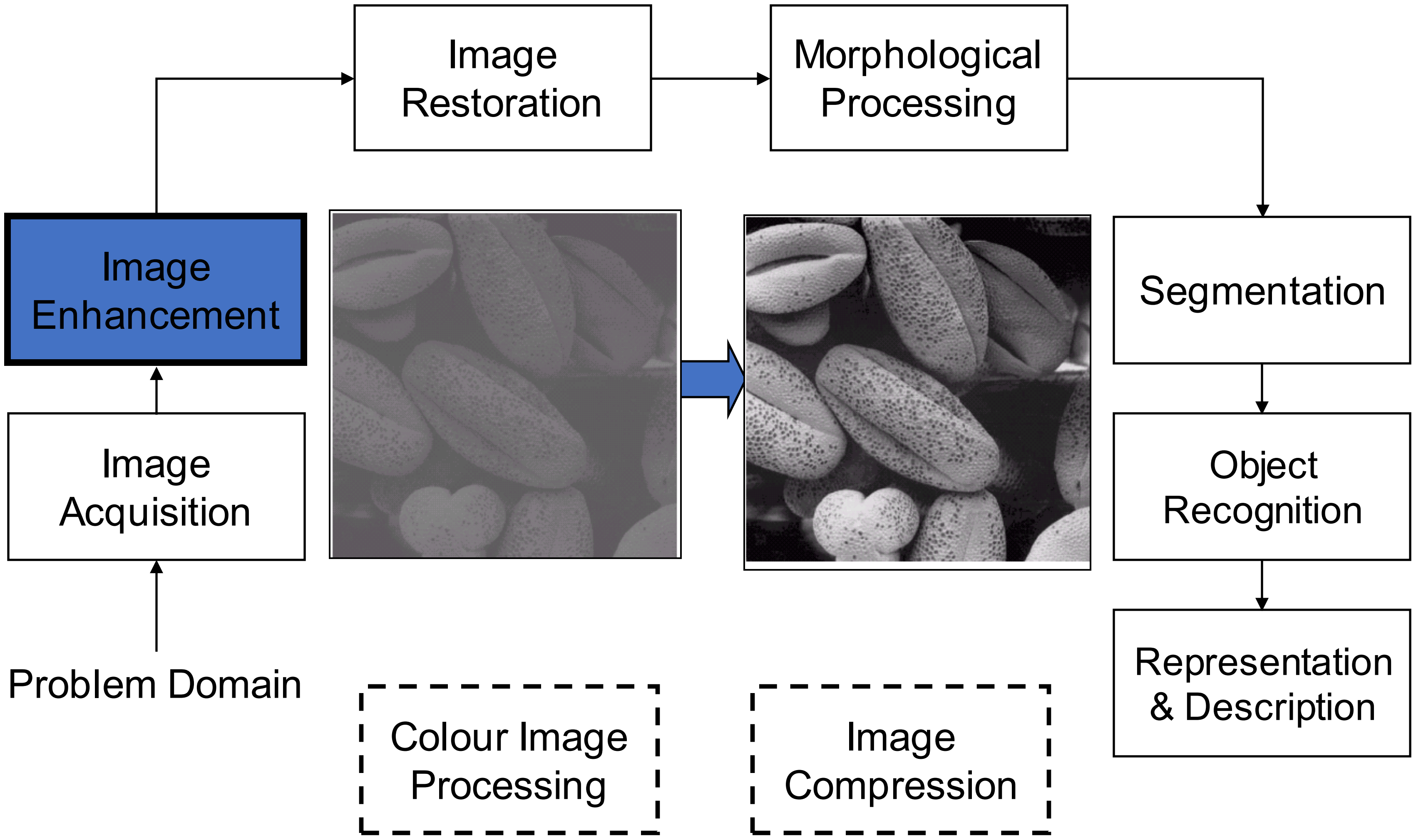
Key Stages in Digital Image Processing



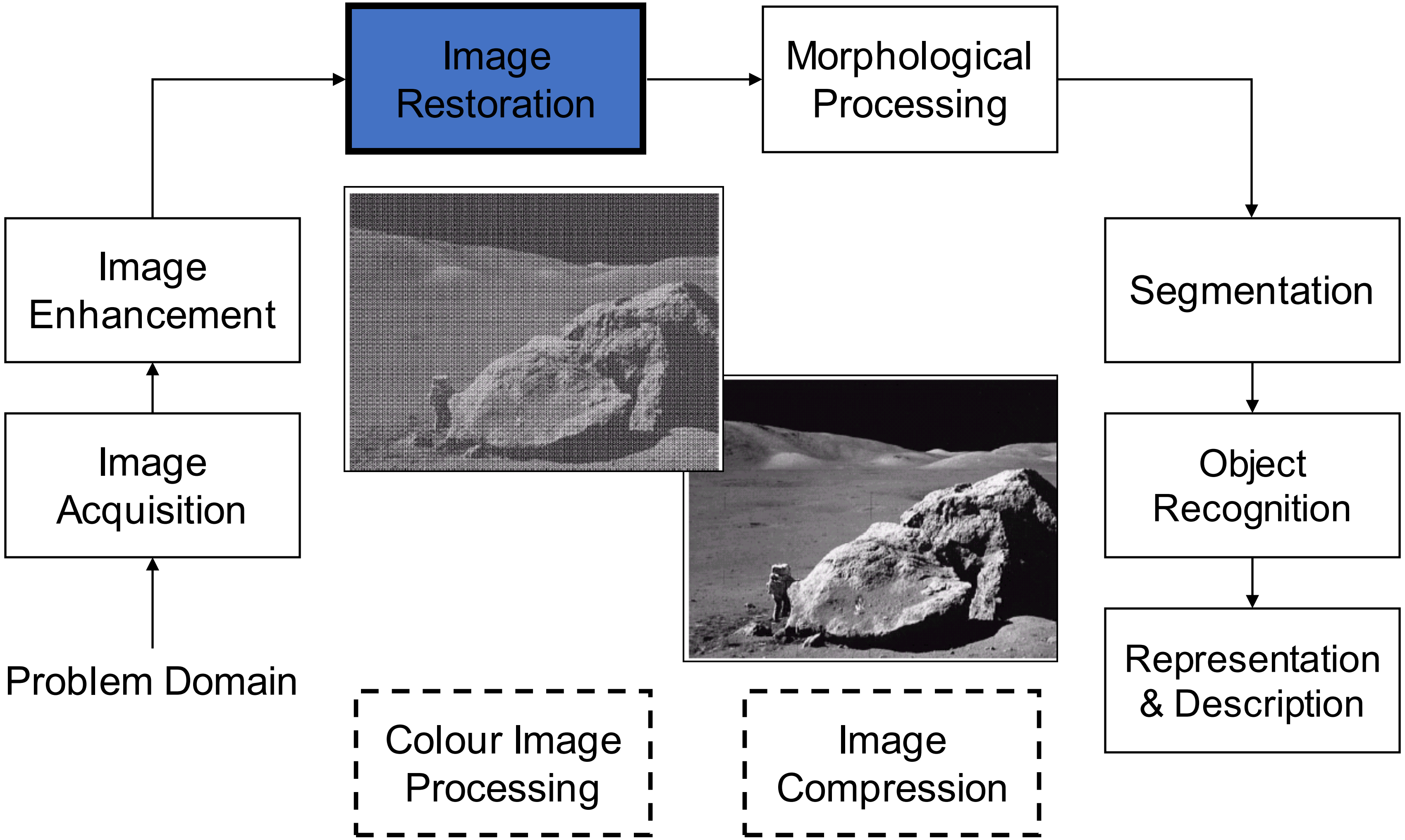
Key Stages in Digital Image Processing



Key Stages in Digital Image Processing



Key Stages in Digital Image Processing



Key Stages in Digital Image Processing

Image Restoration

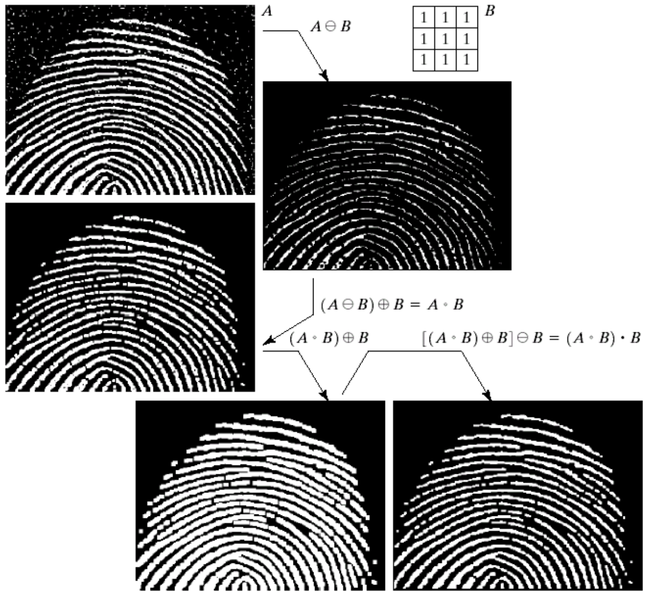
Morphological Processing

erosion followed by a dilation

Image Enhancement

Image Acquisition

Problem Domain



Segmentation

Object Recognition

Representation & Description

Colour Image Processing

Image Compression

Key Stages in Digital Image Processing

Image Restoration

Morphological Processing

Image segmentation is a commonly used technique in digital image processing and analysis to partition an image into multiple parts or regions, often based on the characteristics of the pixels in the image.

Image Enhancement

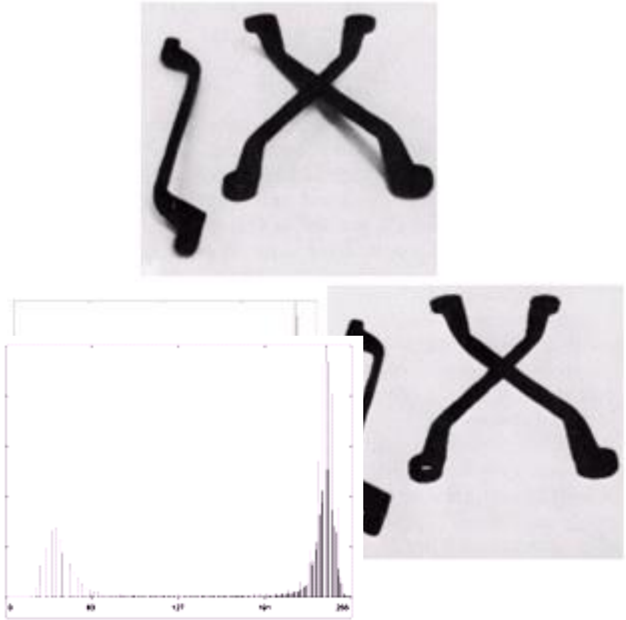
Segmentation

Image Acquisition

Object Recognition

Problem Domain

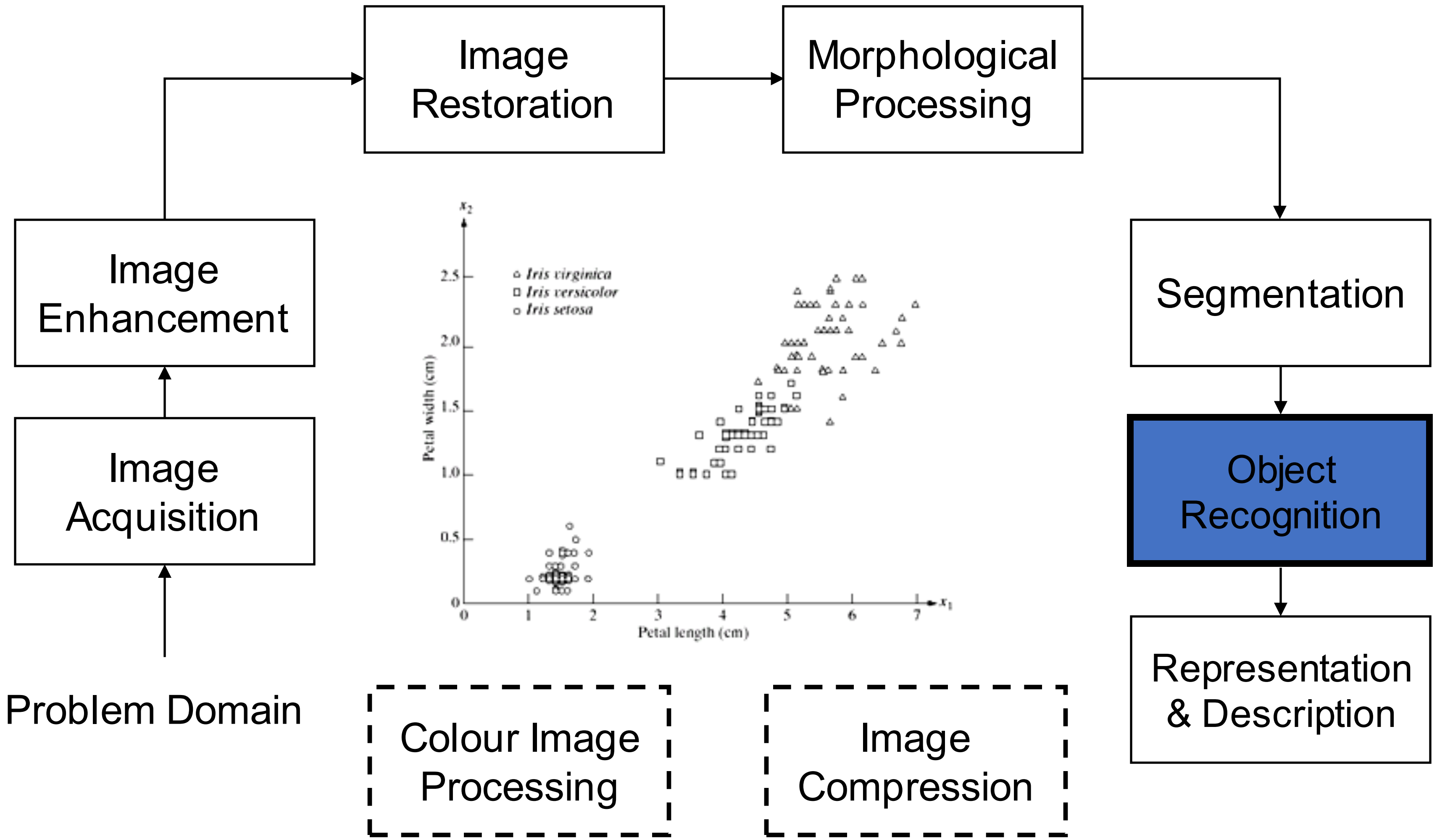
Representation & Description



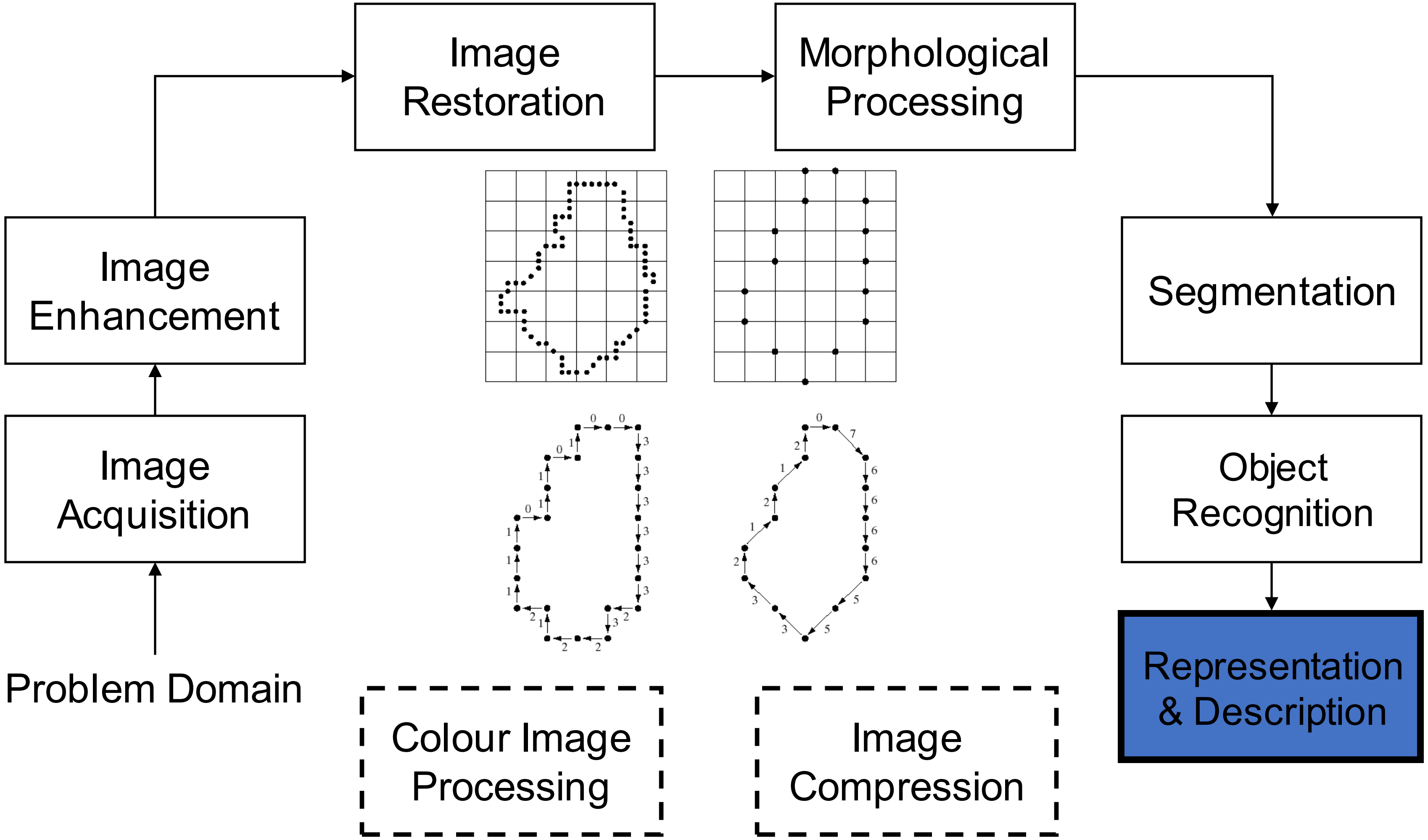
Colour Image Processing

Image Compression

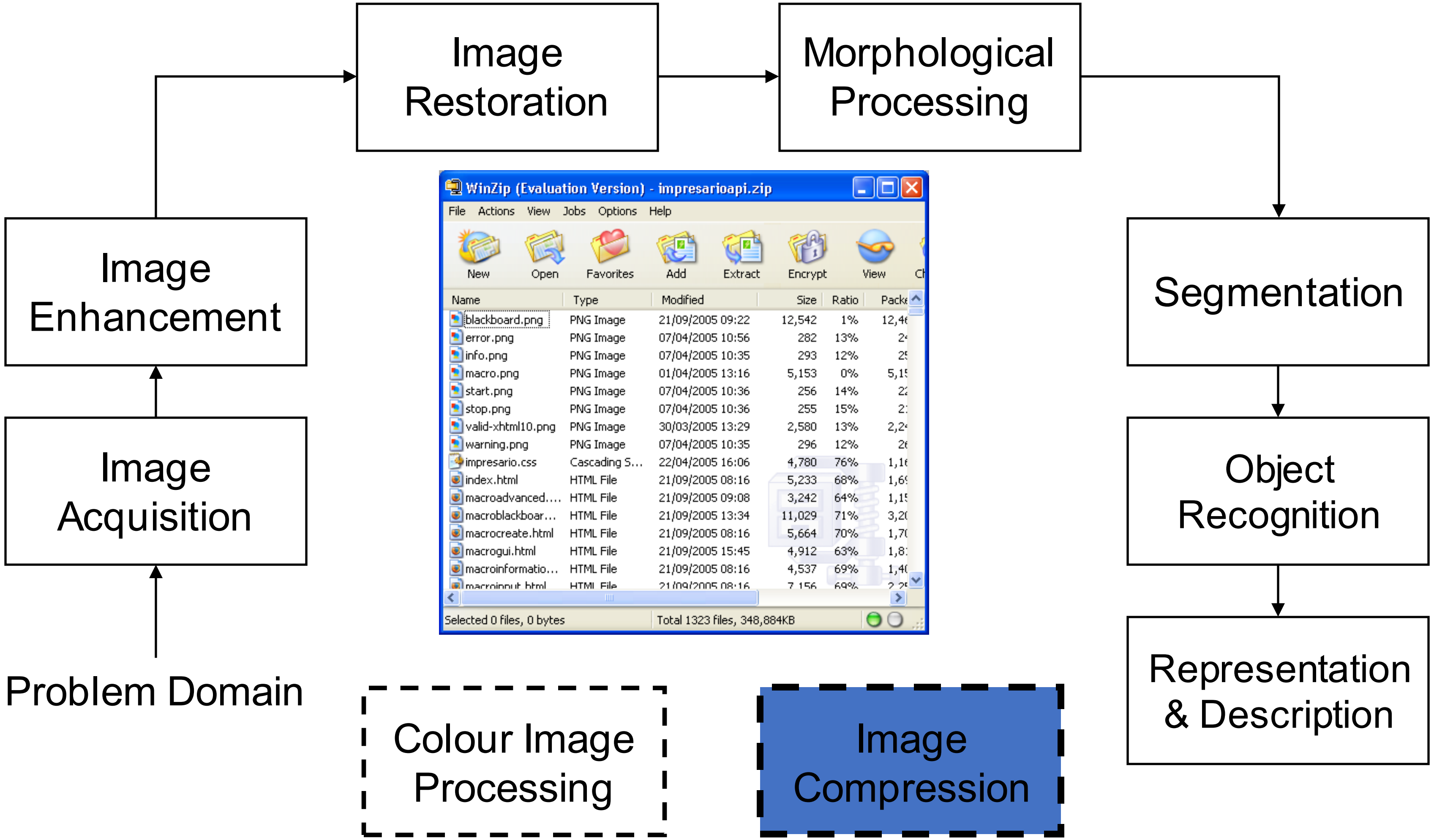
Key Stages in Digital Image Processing



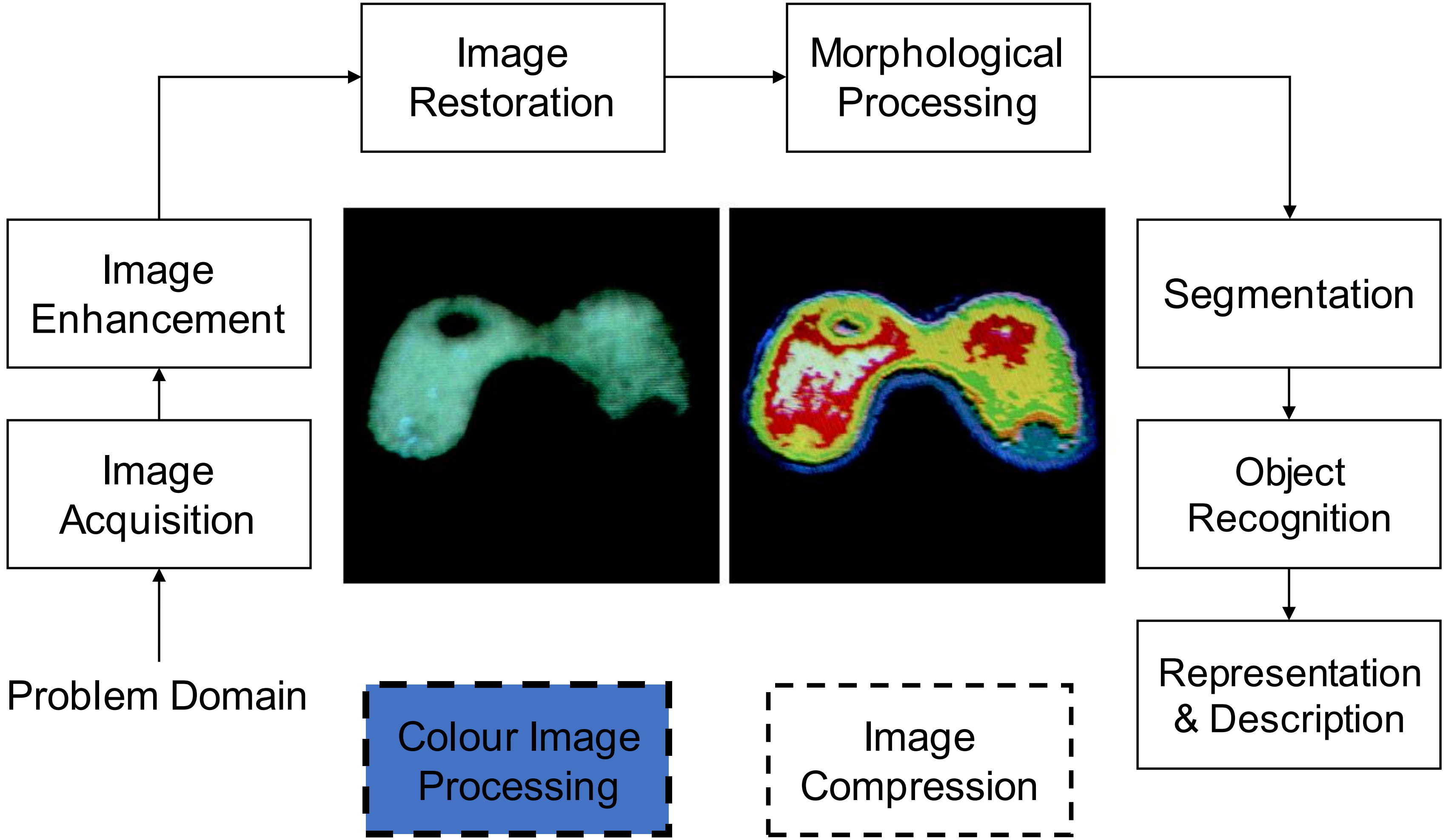
Key Stages in Digital Image Processing



Key Stages in Digital Image Processing



Key Stages in Digital Image Processing



Lecture 5: Sampling and Quantization

A Simple Image Model

- Image: a 2-D light-intensity function $f(x,y)$
- The value of f at (x,y) → the intensity (brightness) of the image at that point
- $0 < f(x,y) < \infty$

Digital Image Acquisition

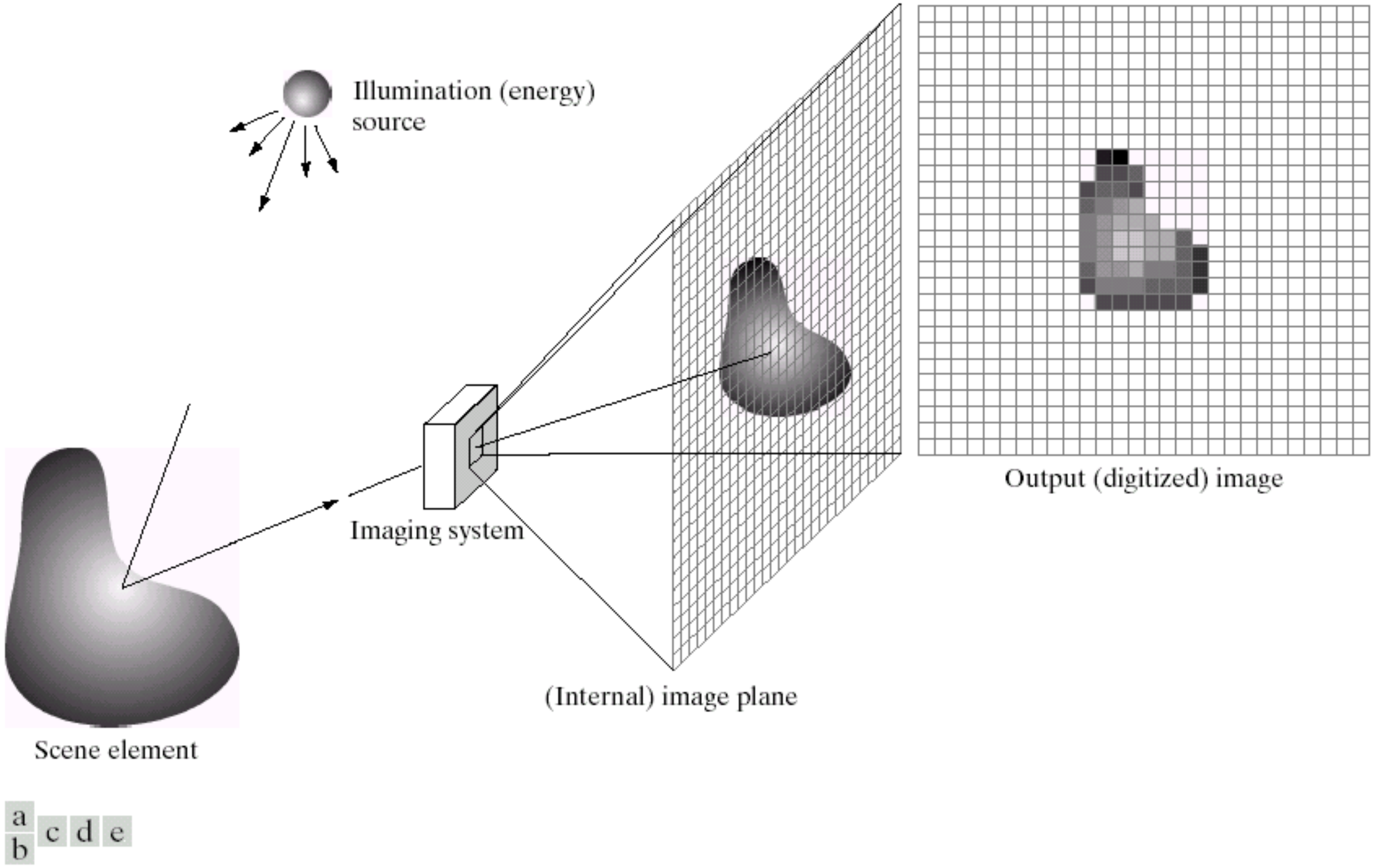


FIGURE 2.15 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

A Simple Image Model

- Nature of f(x,y):
  - The amount of source light incident on the scene being viewed
  - The amount of light reflected by the objects in the scene

A Simple Image Model

- Illumination & reflectance components:
  - Illumination: $i(x,y)$
  - Reflectance: $r(x,y)$

  - $f(x,y) = i(x,y) \cdot r(x,y)$

  - $0 < i(x,y) < \infty$

    and $0 < r(x,y) < 1$

    (from total absorption to total reflectance)

Digital Image

- To create a digital image, we need to convert the continuously sensed data into digital form.
- This involves:
- Sampling:

  Digitizing the coordinate values (resolution)

  Depends on density of sensor in an array

  Limited by optical resolution
- Quantization (bits/pixel)
- Digitizing the amplitude values
- Pixel: short for picture element

Sampling & Quantization

- The spatial and amplitude digitization of f(x,y) is called:
  - image sampling when it refers to spatial coordinates (x,y) and
  - gray-level quantization when it refers to the amplitude.

Digital Image

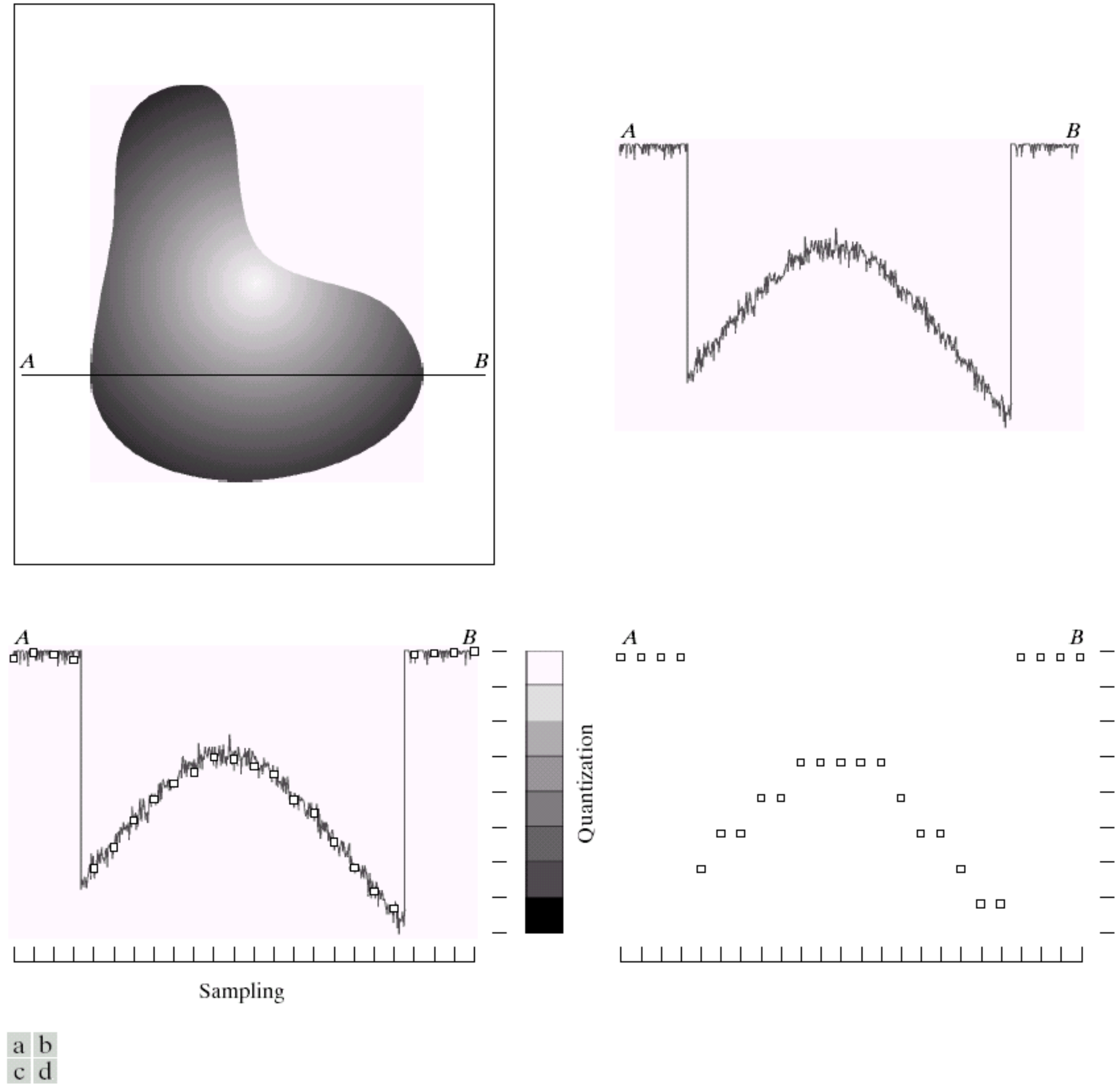
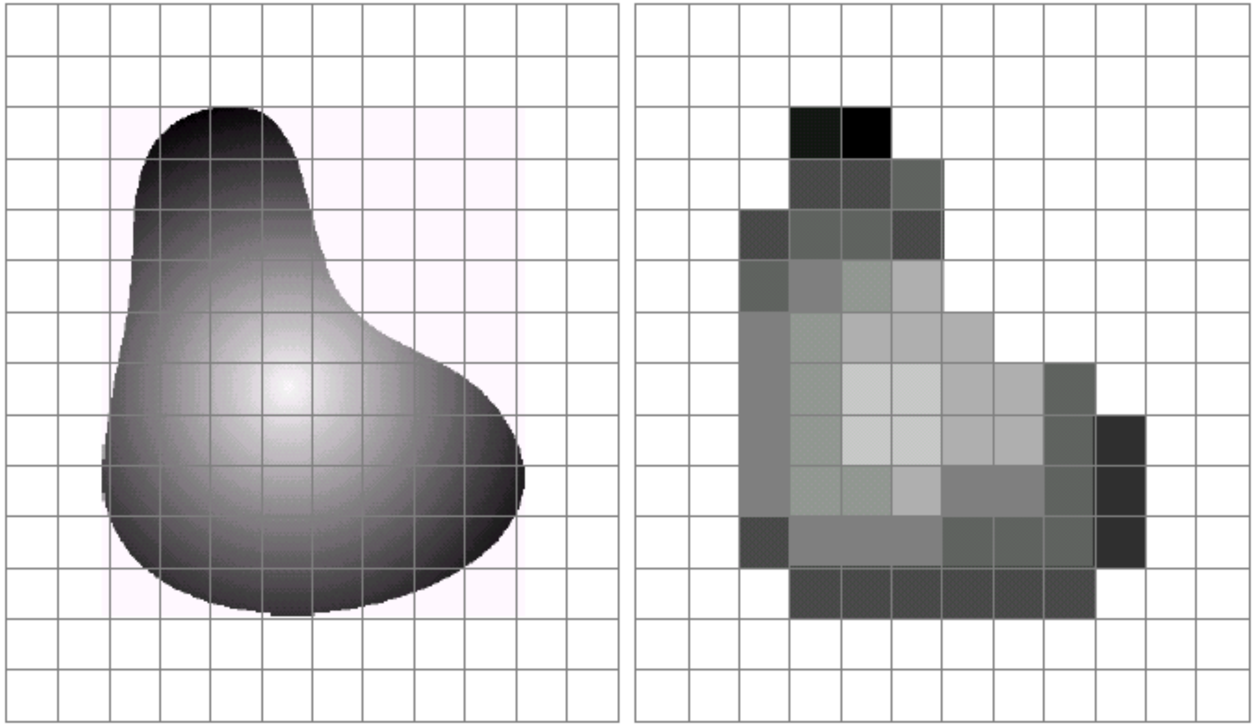


FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

Digital Image

- Digitizing the coordinate values is called **sampling**.
- Digitizing the amplitude values is called **quantization**.

Sampling & Quantization



a b

FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

Digital Image Representation

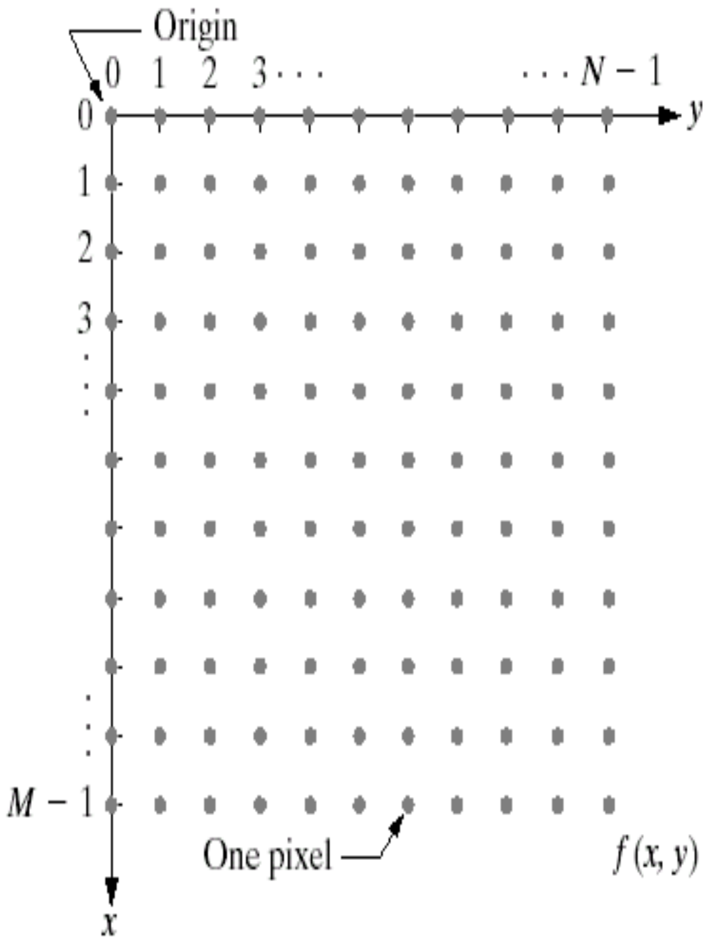


FIGURE 2.18 Coordinate convention used in this book to represent digital images.

Digital Image Representation

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & \dots & \dots & f(1,M-1) \\ \dots & \dots & \dots & \dots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix}$$

Digital Image

Image Elements
(Pixels)

Spatial and Intensity Resolution

- **Spatial resolution** - measure of the smallest discernible detail in an image.
  - line pairs per unit distance, and dots (pixels) per unit distance

- **Intensity Resolution** - smallest discernible change in intensity level.
  - the number of intensity levels usually is an integer power of two

Example

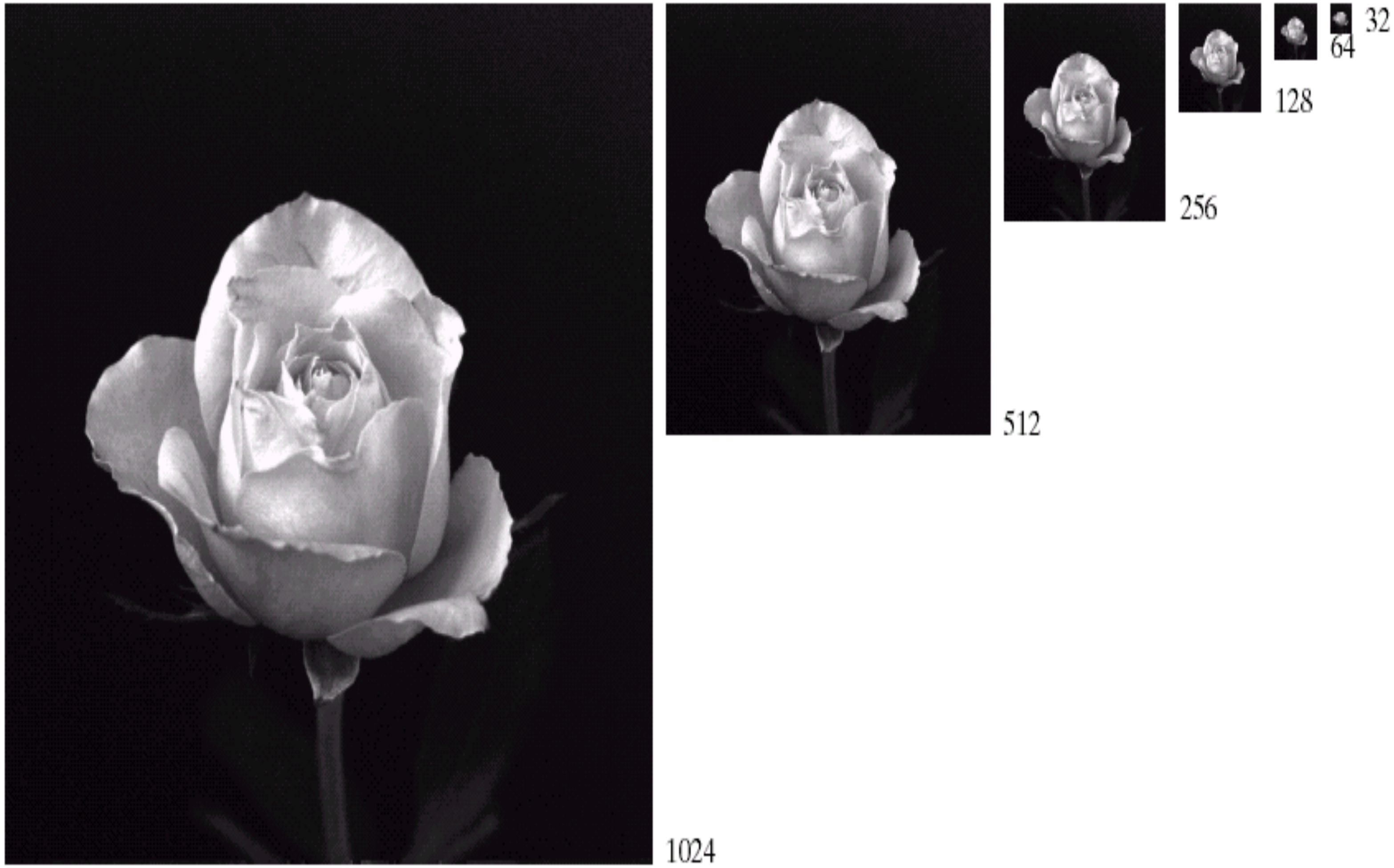
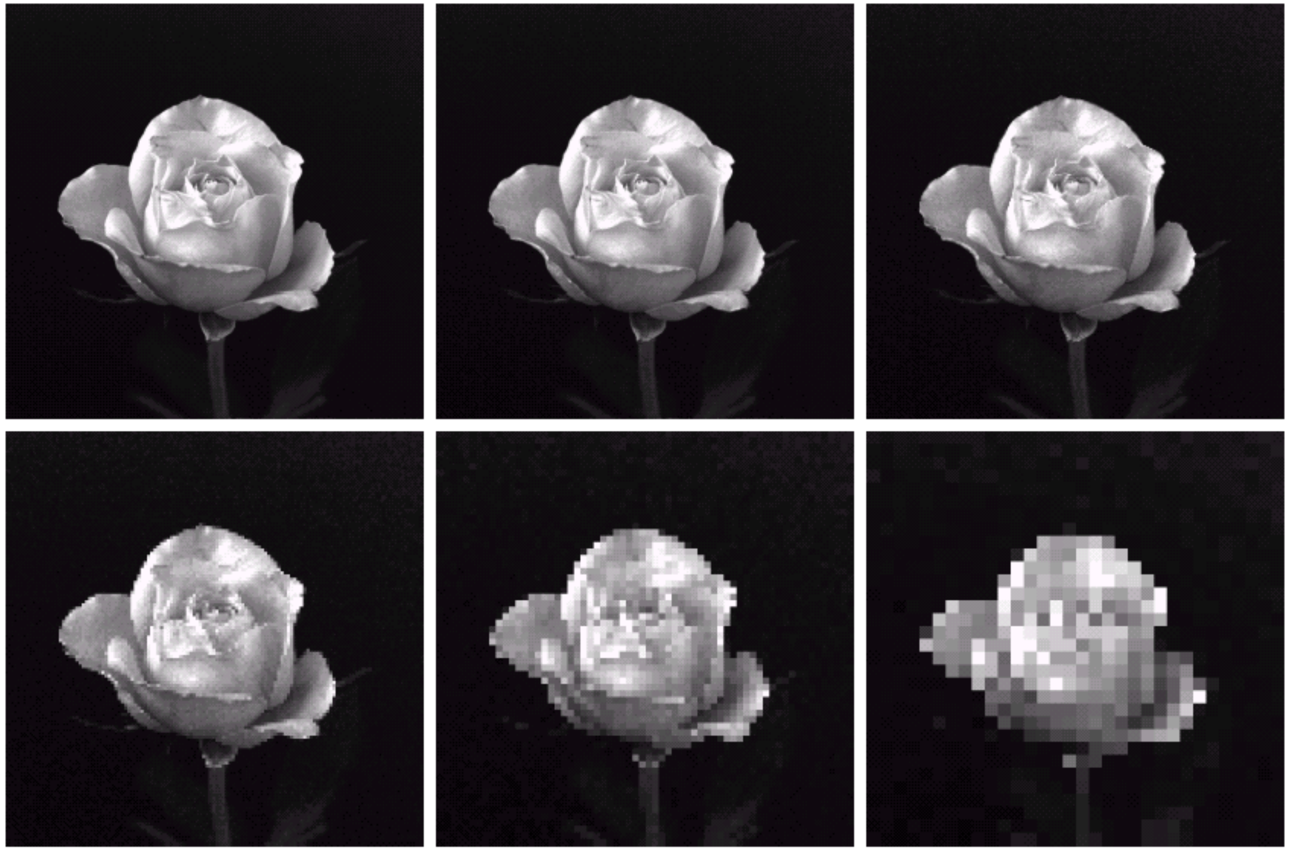


FIGURE 2.19 A 1024×1024, 8-bit image subsampled down to size 32×32 pixels. The number of allowable gray levels was kept at 256.

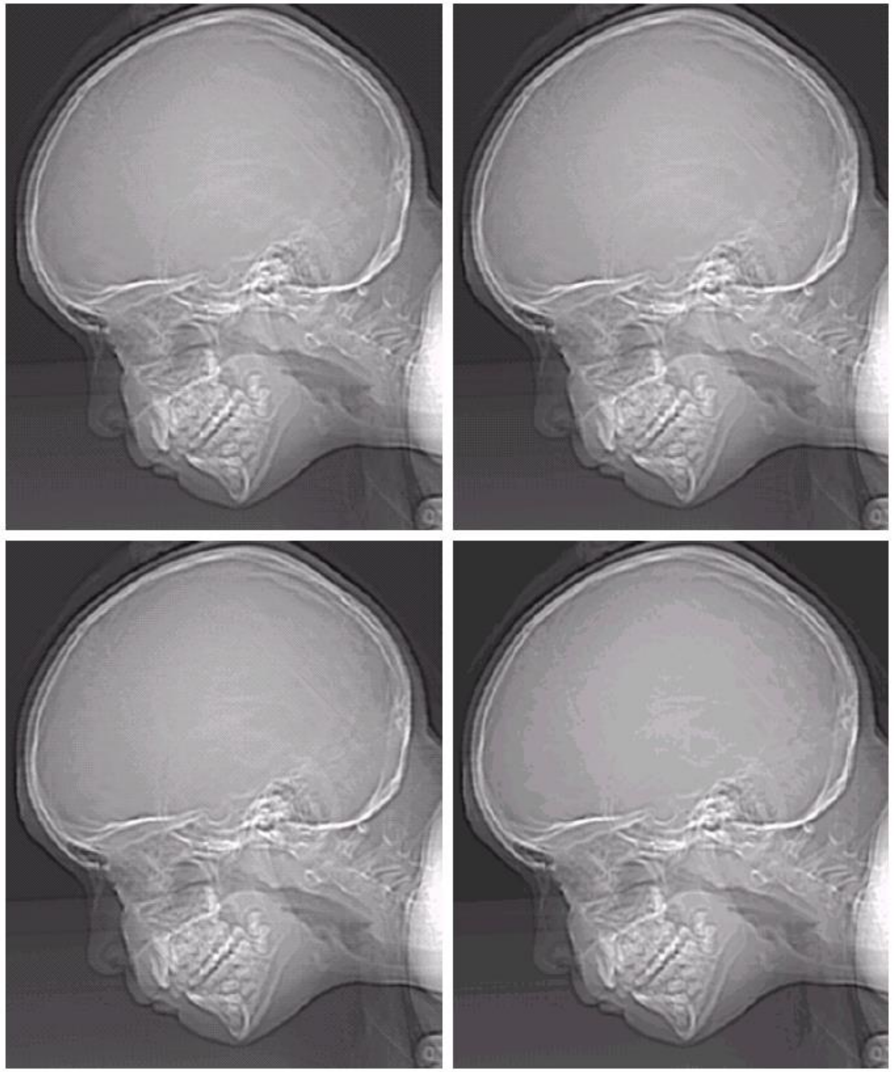
Examples



a b c
d e f

FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

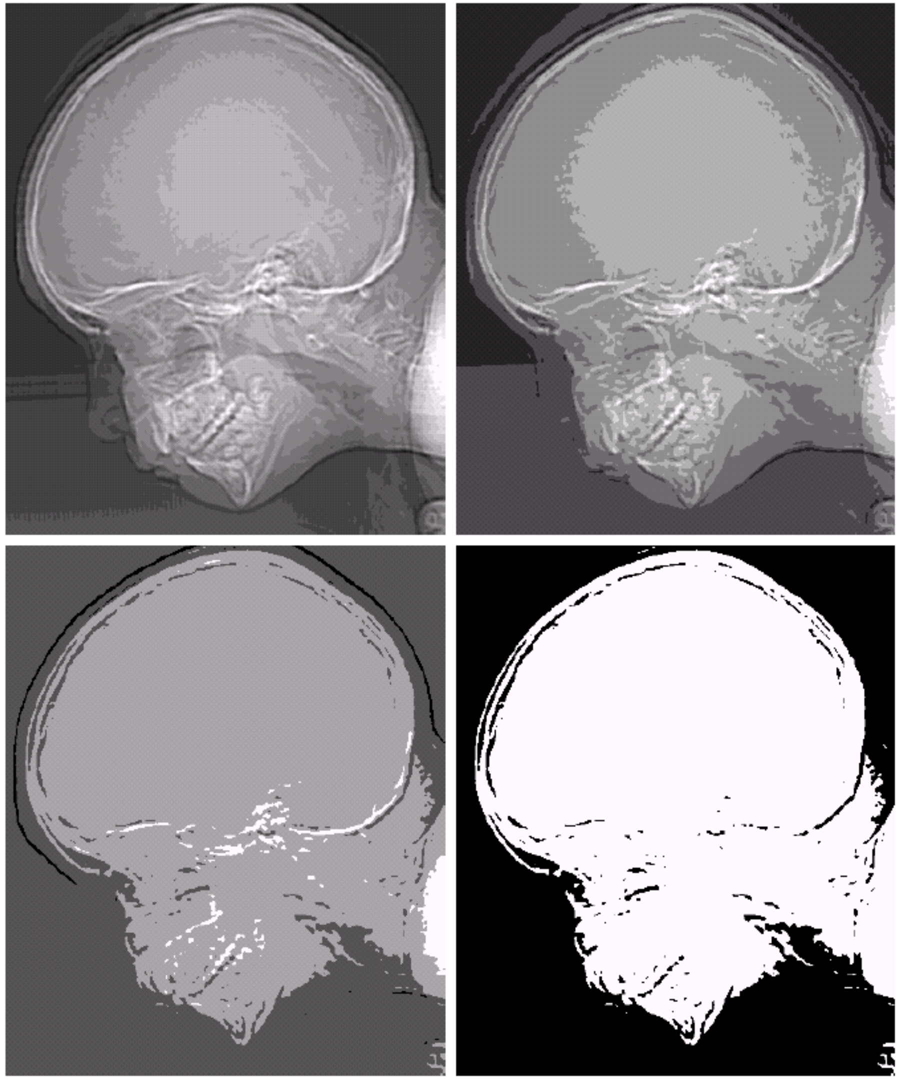
Examples



a b
c d

FIGURE 2.21
(a) 452 × 374, 256-level image. (b)–(d) Image displayed in 128, 64, and 32 gray levels, while keeping the spatial resolution constant.

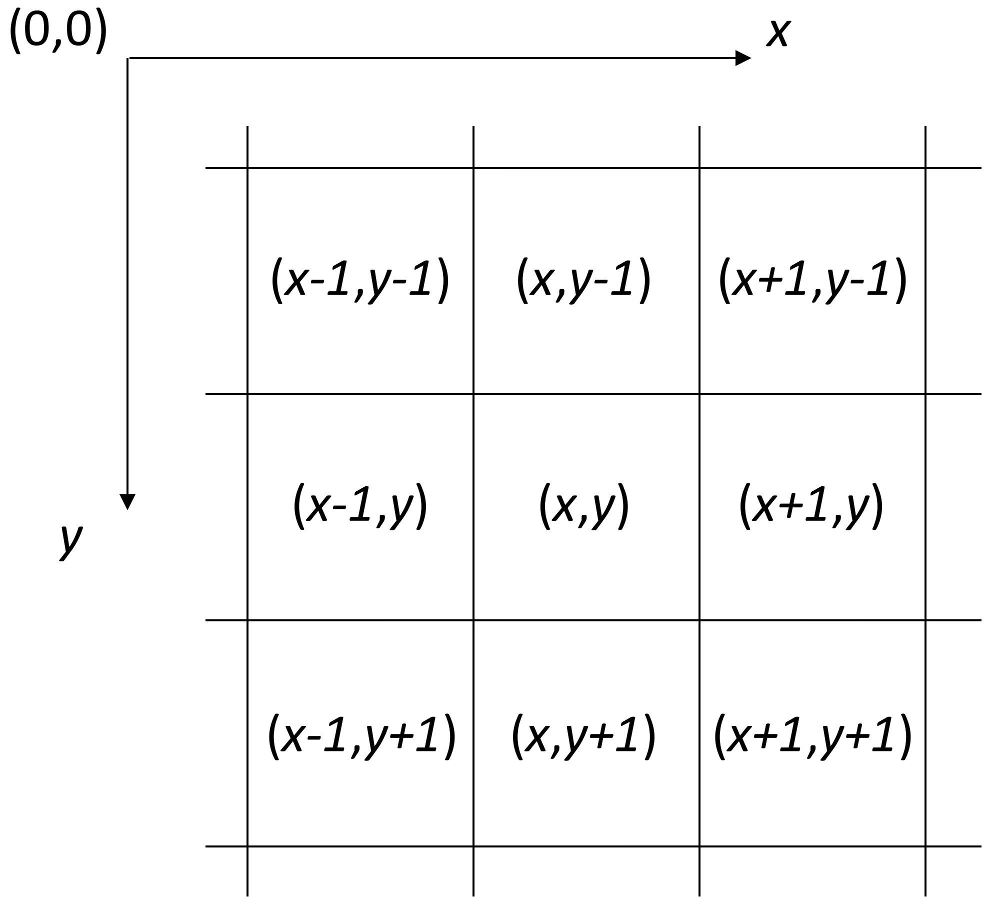
Examples



e f
g h

FIGURE 2.21 *(Continued)* (e)–(h) Image displayed in 16, 8, 4, and 2 gray levels. (Original courtesy of Dr. David R. Pickens, Department of Radiology & Radiological Sciences, Vanderbilt University Medical Center.)

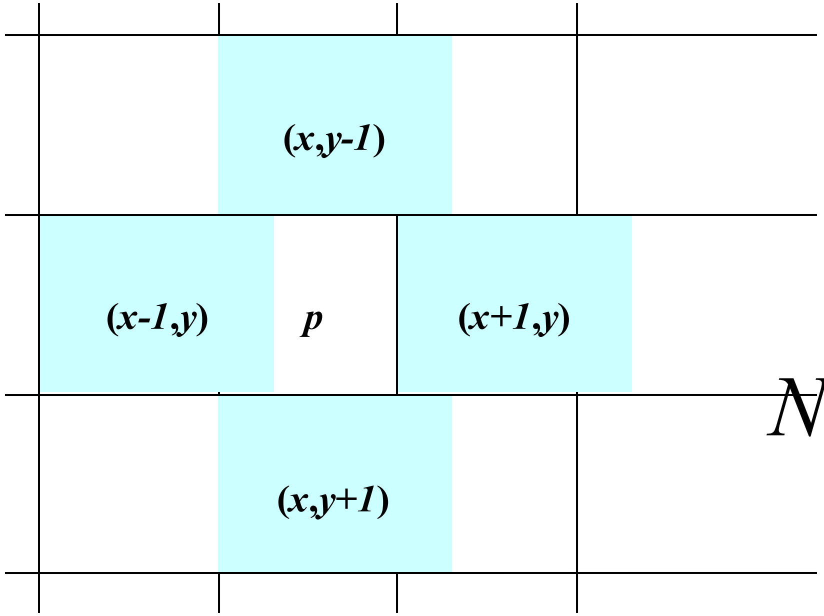
Basic Relationship of a Pixel



Conventional indexing method

Neighbors of a Pixel

Neighborhood relation is used to tell adjacent pixels. It is useful for analyzing regions.



4-neighbors of *p*:

$$N_4(p) = \left\{ \begin{array}{l} (x-1,y) \\ (x+1,y) \\ (x,y-1) \\ (x,y+1) \end{array} \right\}$$

4-neighborhood relation considers only vertical and horizontal neighbors.

Note: q Î $N_4(p)$ implies p Î $N_4(q)$

Neighbors of a Pixel

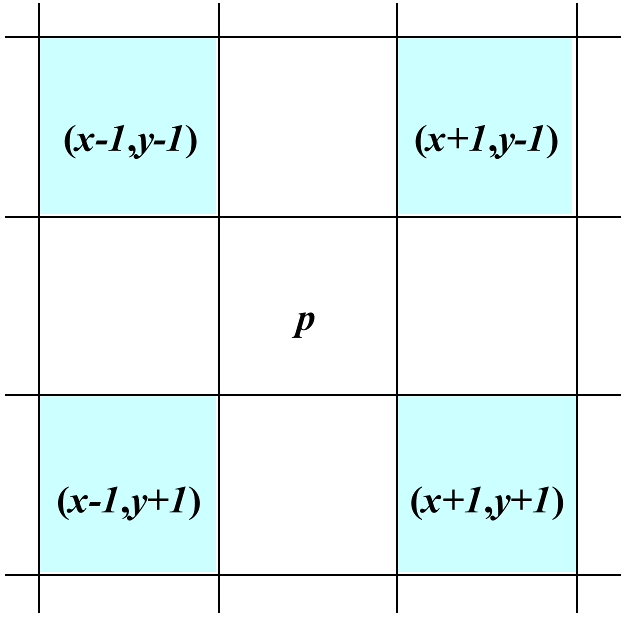
(x-1,y-1)	***(x,y-1)***	***(x+1,y-1)***
(x-1,y)	***p***	***(x+1,y)***
(x-1,y+1)	***(x,y+1)***	***(x+1,y+1)***

8-neighbors of *p*:

$$N_8(p) = \left\{ \begin{array}{c} (x-1,y-1) \\ (x,y-1) \\ (x+1,y-1) \\ (x-1,y) \\ (x+1,y) \\ (x-1,y+1) \\ (x,y+1) \\ (x+1,y+1) \end{array} \right\}$$

8-neighborhood relation considers all neighbor pixels.

Neighbors of a Pixel



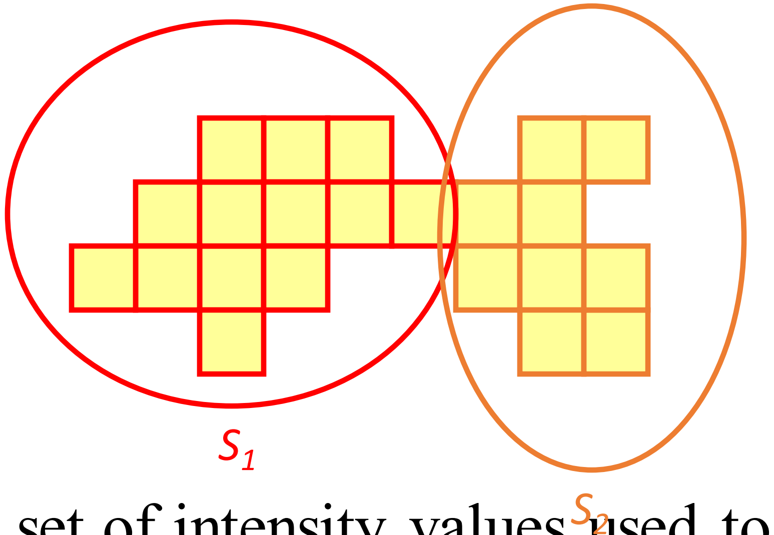
Diagonal neighbors of *p*:

$$N_D(p) = \left\{ \begin{array}{c} (x-1,y-1) \\ (x+1,y-1) \\ (x-1,y+1) \\ (x+1,y+1) \end{array} \right\}$$

Diagonal -neighborhood relation considers only diagonal neighbor pixels.

Adjacency

- Two image subsets S_1 and S_2 are adjacent if some pixel in S_1 is adjacent to some pixel in S_2



- Let V be the set of intensity values used to define adjacency.
- In a binary image, $V = \{1\}$ if we are referring to adjacency of pixels with value 1.

Types of Adjacency

1. **4-adjacency:** Two pixels p and q with values from V are 4-adjacent if q is in the set $N_4(p)$.

2. **8-adjacency:** Two pixels p and q with values from V are 8-adjacent if q is in the set $N_8(p)$.

3. **m-adjacency =** Two pixels are said to be mixed adjacent if
   1. ***q* is in *N*4(*p*), *or***
   2. q is in $ND(p)$ *and* the set $N4(p)$¨$N4(q)$ has no pixels whose values are from V.

Connectivity

- A *digital path* (or *curve*) from pixel *p* with coordinates (x_0,y0) to pixel *q* with coordinates (x_n , y_n) is a sequence of distinct pixels with coordinates

$$(x_0,y_0),(x_1,y_1),...,(x_n,y_n)$$

- Let *S* represent a subset of pixels in an image. Two pixels *p* and *q* are said to be ***connected*** *in S* if there exists a path between them consisting entirely of pixels in *S*.

- For any pixel *p* in *S*, the set of pixels that are connected to it in *S* is called a *connected component* of *S*.

Lecture 7: Discrete Fourier Transform

Fourier Transform

- An image can be transformed to hide or show information.
- Image Transformation – spatial domain and frequency domain.
- Spatial Domain – changing the values of pixels based on certain constraints.
  - Change brightness and clarity of the image

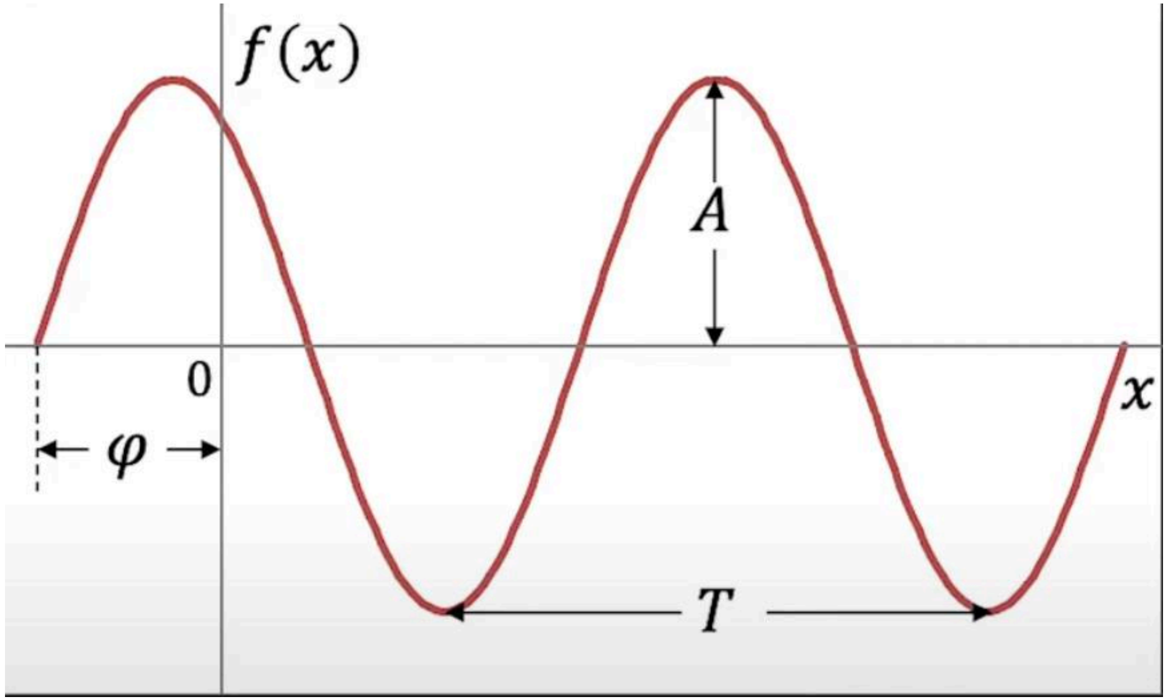
Fourier Transform

- Frequency Domain – frequency content in the image.
  - Represent image in more compact form.
  - Computationally efficient to store and transmit images.
  - Separate noise and salient information.

- Any periodic function can be rewritten as weighted sum of infinite sinusoids of different frequencies.

Sinusoid

- $f(x) = Asin(2\pi ux + \varphi)$

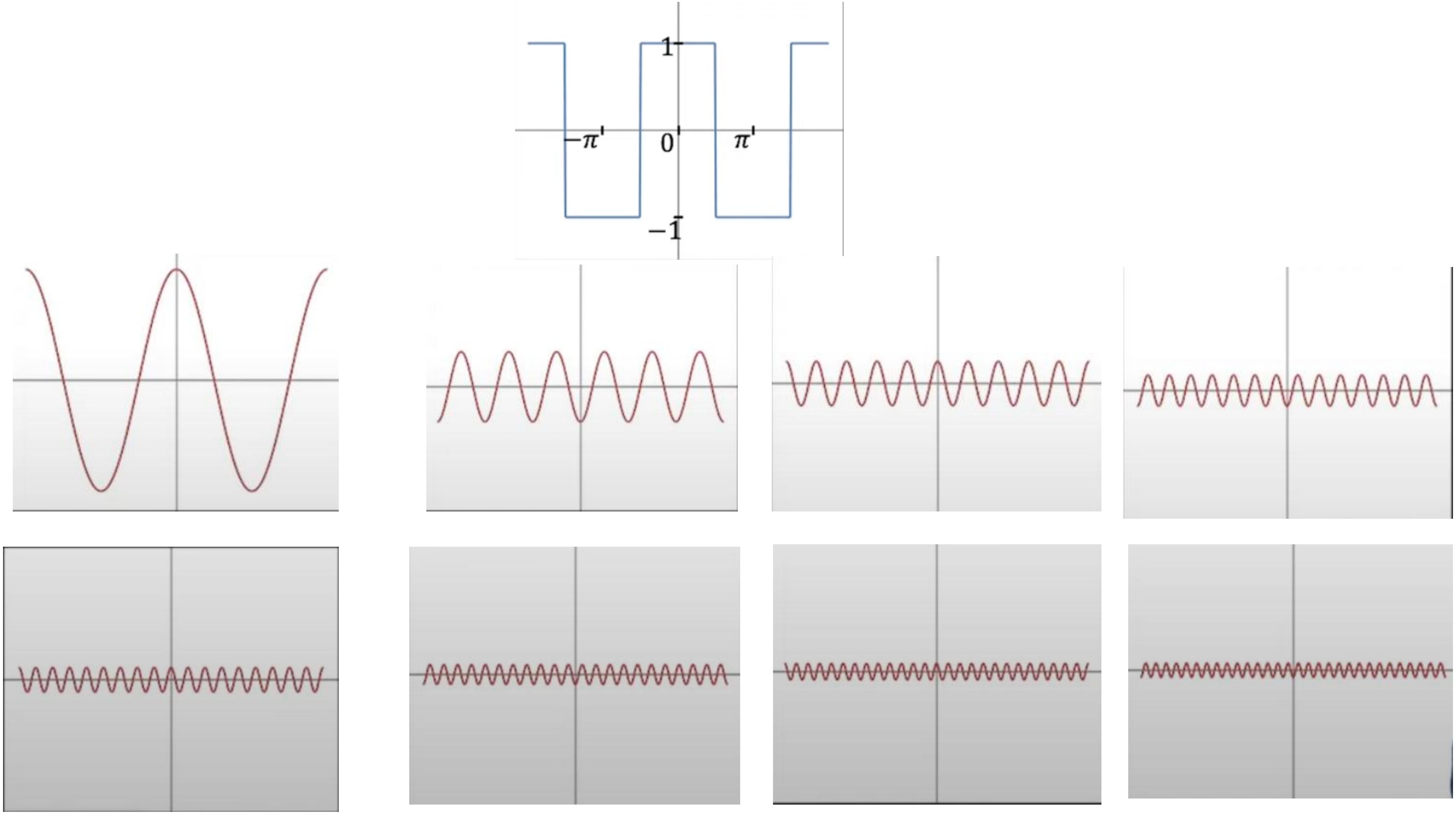


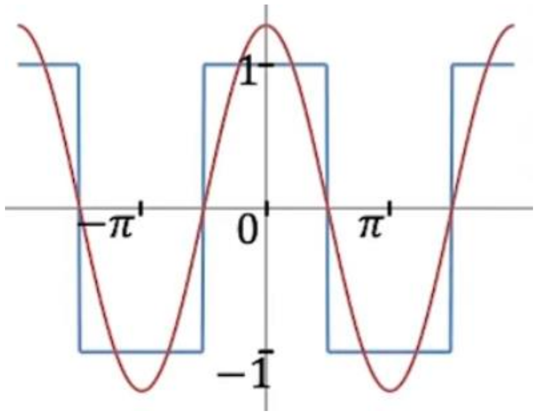
A: Amplitude

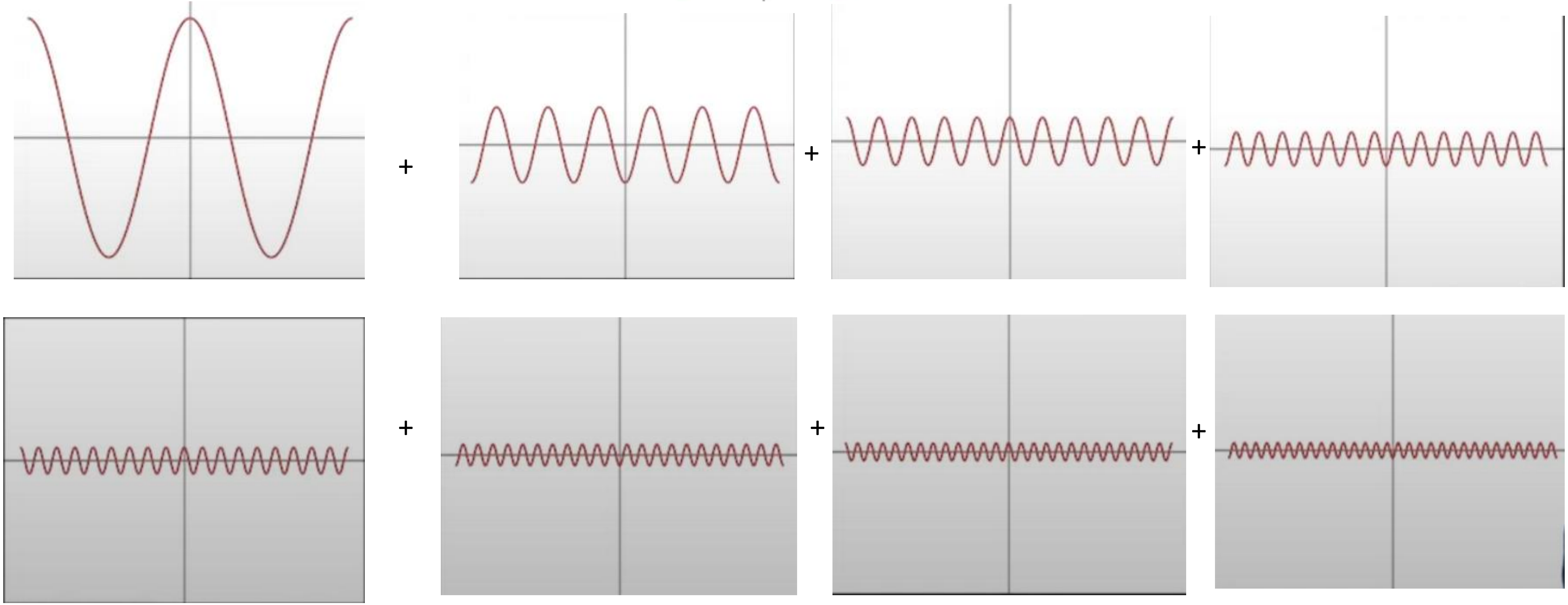
φ: Phase

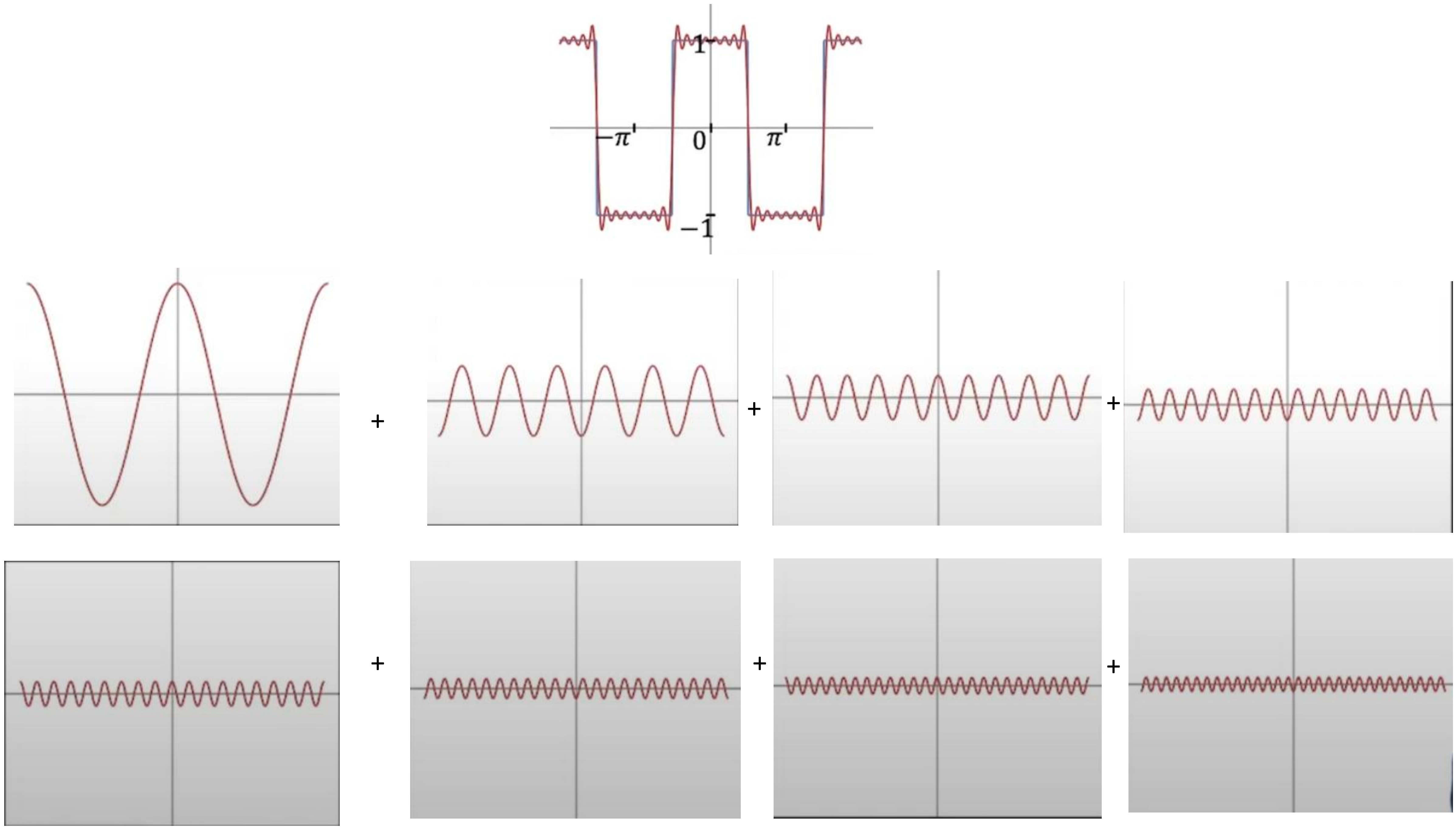
T: Period

u: frequency (1/T)


1
−π
0
π
−1

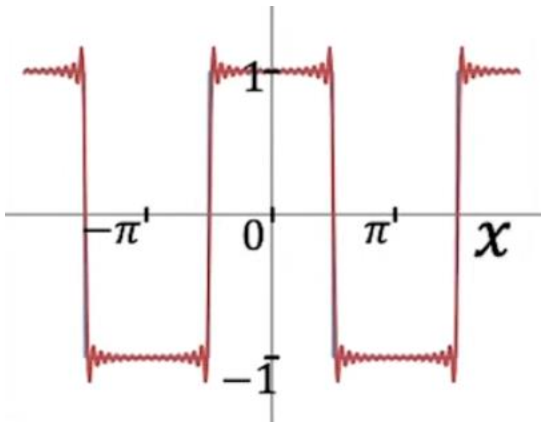

1
−π
0
π
−1

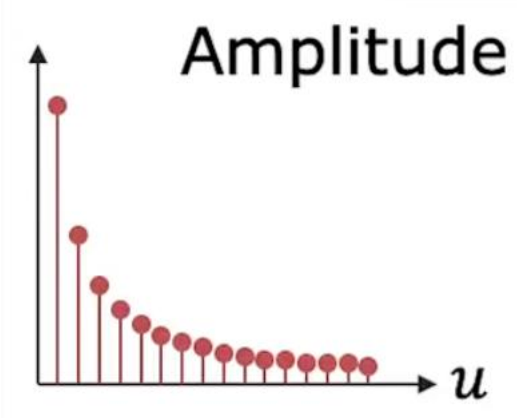

+
+
+
+
+
+

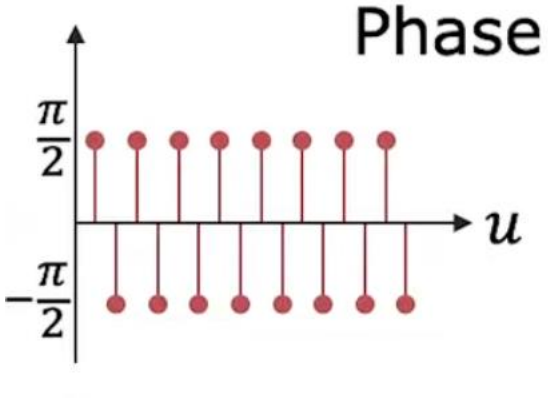


1
−π
0
π
−1
+
+
+
+
+
+

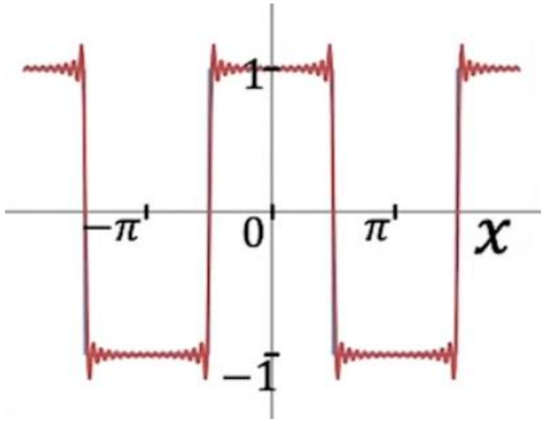
Frequency Representation of Signal

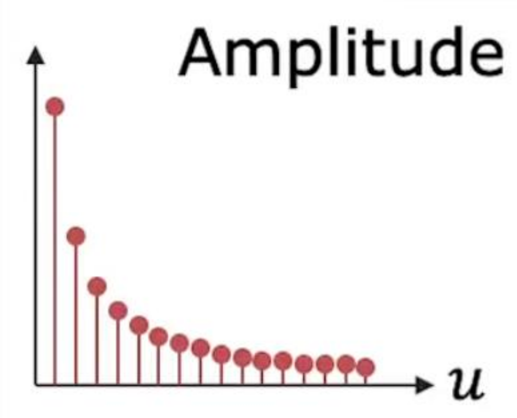




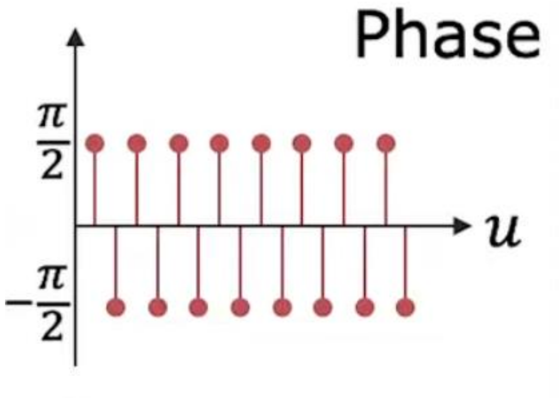


Frequency Representation of Signal





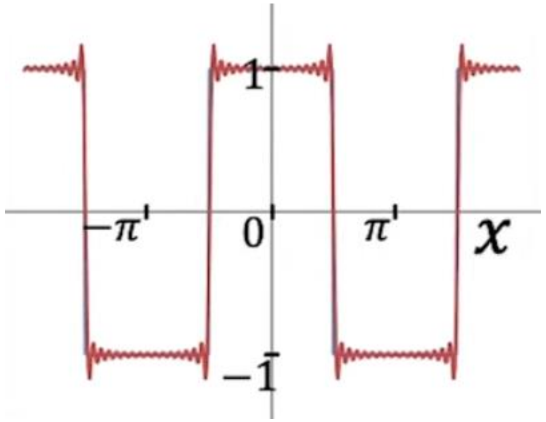
+



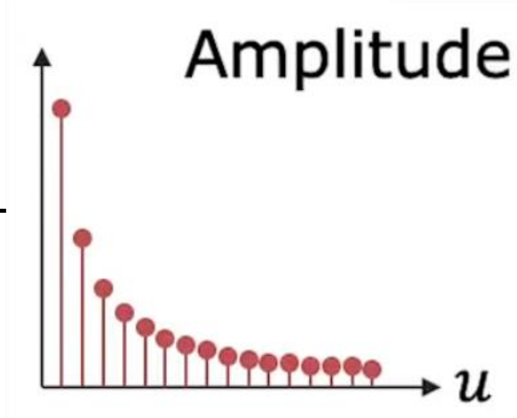
- Represents a signal f(x) in terms of Amplitude and phases of its constituent sinusoids.

f(x) ⟶ FT ⟶ f(u)

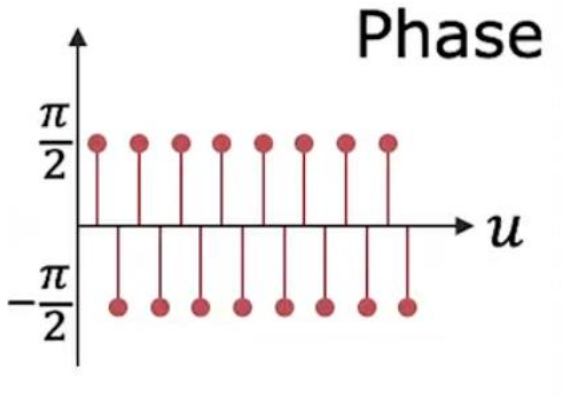
Inverse Fourier Transform



←



+



- Represents a signal f(x) in terms of Amplitude and phases of its constituent sinusoids.

f(x) ⟶ FT ⟶ f(u)

Discrete Fourier Transform

- Decompose an image (2D signal) into cosine and sine components.

- Linear combination of sine and cosine waves of different frequencies.

- Consider a 1D function,{f(x), $0 \leq x \leq N-1$}. The general form of a transformation is

$$g(u) = \sum_{x=0}^{N-1} T(u,x) f(x); \qquad 0 \leq u \leq N-1$$

- T(u,x): forward kernel, g(u): transformed image.

Discrete Fourier Transform

- If the transformation is Discrete Fourier Transform (DFT)-

$$g(u) = \sum_{x=0}^{N-1} \frac{1}{N} e^{-i2\pi\frac{x}{N}} f(x); u = 0,1,2,\ldots N-1$$

- The inverse DFT will be

$$f(x) = \sum_{x=0}^{N-1} e^{i2\pi\frac{ux}{N}} g(u)$$

Discrete Fourier Transform – 2D Image

- Consider an image f(x,y)of size MxN. The 2-D DFT of f(x,y) is defined as follows:

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

- Inverse 2D DFT is given by:

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

Discrete Fourier Transform – 2D Image

- This difference is called the “phase angle”.

- Phase Information - recovering the original information, edge information, boundary information.

Discrete Fourier Transform – 2D Image

- Representation of intensity as a function of frequency : Spectrum
- The coordinates of the Fourier spectrum: spatial frequencies.
- The spatial position information of an image is encoded as the difference between the coefficients of the real and imaginary parts.

Discrete Fourier Transform – 2D Image

- image values f (x, y) – real

- the corresponding frequency domain data - complex.

- Result: one matrix containing real values R(u,v) and the other matrix I(u,v) will contain the imaginary component of the complex value.

- The amplitude spectrum or the magnitude for 2D DFT is given by

$$|F(u,v)| = [R^2(u,v) + I^2(u,v)]^{1/2}$$

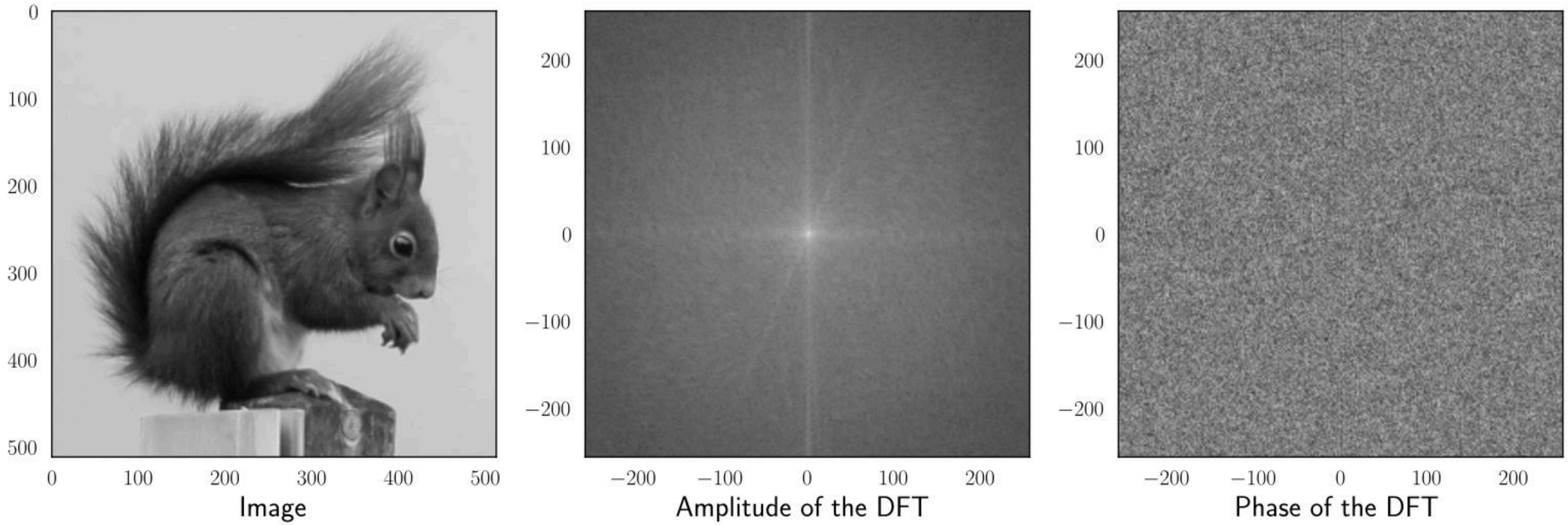
DFT Properties

- If f (x, y) is real and even then F(u, v) is real and even.
- If f (x, y) is real and odd then F(u, v) is imaginary and odd.
- Distributive property
- Rotation: Rotating f (x, y) by <angle> rotates F(u, v) by <angle>.

DFT Properties

Property	
Separability	$T(u, x, v, y) = T_1(u, x).T_2\ (v, y)$
Symmetry	$T_1(u, x)=T_2\ (u, x)$
Periodicity	$F(u, v)=F(u+ M,v)=F(u, v+N)=F(u+M,\ v+N\ )$
Conjugate Symmetry	$F(u, v)\ =F^*(-u+pM,\ -v+qN)$

Applications of DFT



Discrete Fourier Transform – 2D Image

- The power spectrum of 2D-DFT

$$P(u,v) = \left|F(u,v)\right|^2 = R^2(u,v) + I^2(u,v)$$

- Phase spectrum of 2D-DFT

$$\text{ø}(u,v) = \tan^{-1}\frac{I(u,v)}{R(u,v)}$$

Applications of DFT