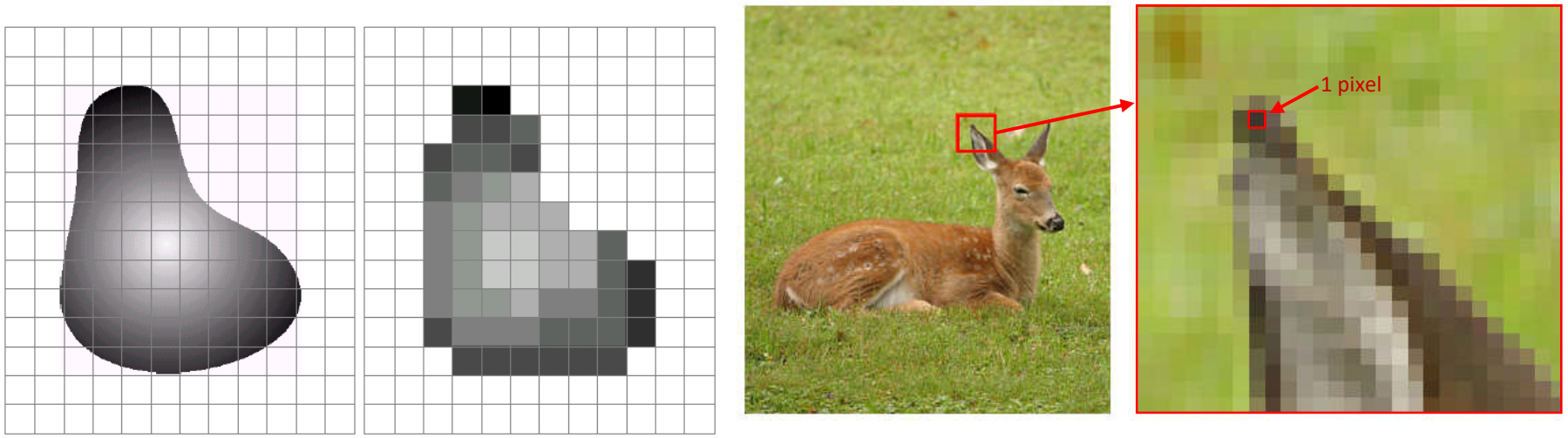
# Image Formation Fundamentals

Representation, sampling,
quantization, formation, convolution

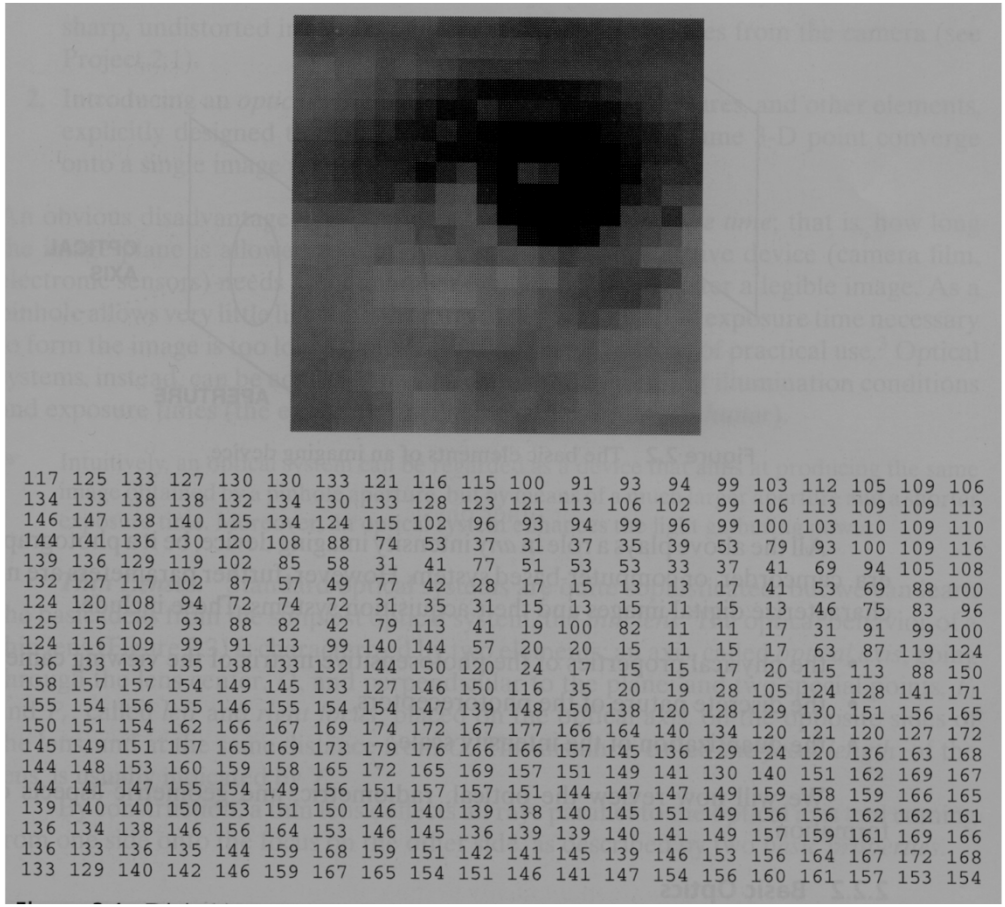
- What is the role of image processing in object recognition or object detection?

# 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

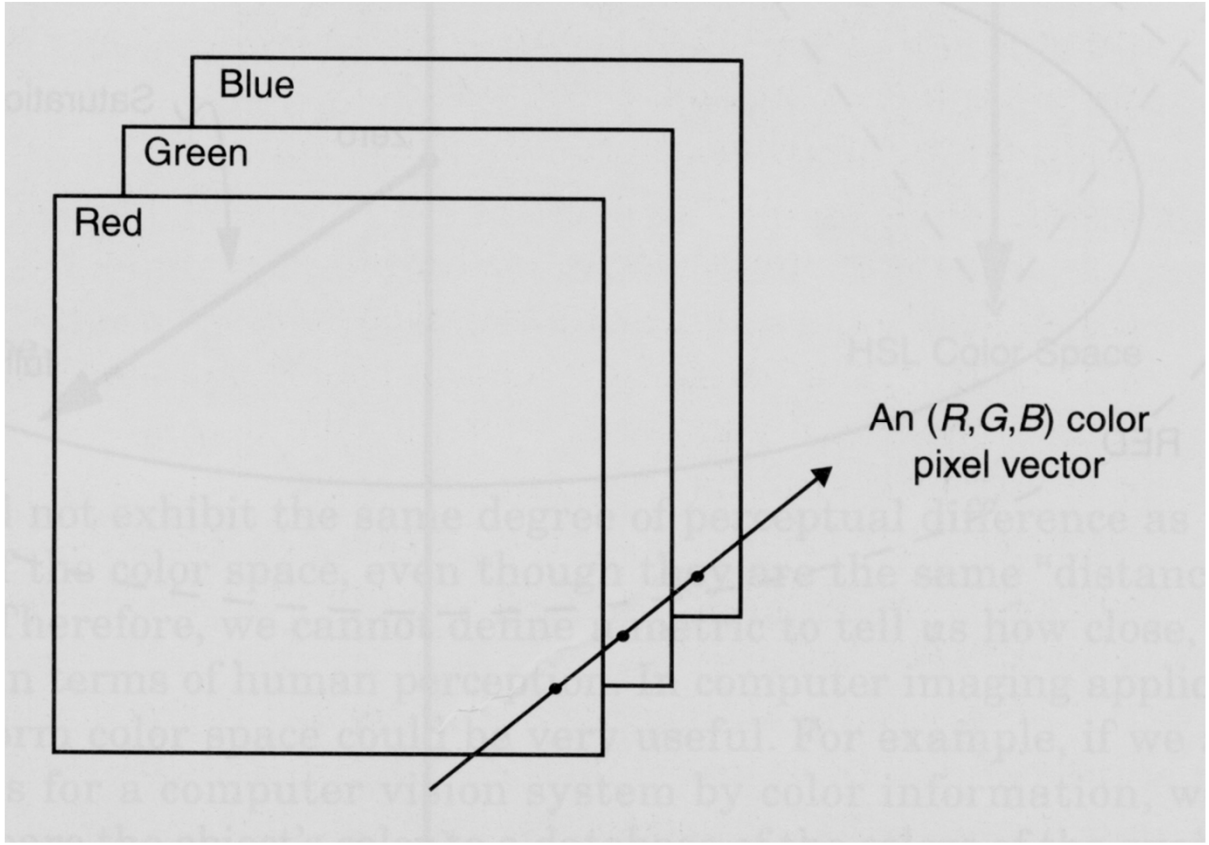


# How are images represented in the computer?



| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 117 | 125 | 133 | 127 | 130 | 130 | 133 | 121 | 116 | 115 | 100 | 91 | 93 | 94 | 99 | 103 | 112 | 105 | 109 | 106 |
| 134 | 133 | 138 | 138 | 132 | 134 | 130 | 133 | 128 | 123 | 121 | 113 | 106 | 102 | 99 | 106 | 113 | 109 | 109 | 113 |
| 146 | 147 | 138 | 140 | 125 | 134 | 124 | 115 | 102 | 96 | 93 | 94 | 99 | 96 | 99 | 100 | 103 | 110 | 109 | 110 |
| 144 | 141 | 136 | 130 | 120 | 108 | 88 | 74 | 53 | 37 | 31 | 37 | 35 | 39 | 53 | 79 | 93 | 100 | 109 | 116 |
| 139 | 136 | 129 | 119 | 102 | 85 | 58 | 31 | 41 | 77 | 51 | 53 | 53 | 33 | 37 | 41 | 69 | 94 | 105 | 108 |
| 132 | 127 | 117 | 102 | 87 | 57 | 49 | 77 | 42 | 28 | 17 | 15 | 13 | 13 | 17 | 41 | 53 | 69 | 88 | 100 |
| 124 | 120 | 108 | 94 | 72 | 74 | 72 | 31 | 35 | 31 | 15 | 13 | 15 | 11 | 15 | 13 | 46 | 75 | 83 | 96 |
| 125 | 115 | 102 | 93 | 88 | 82 | 42 | 79 | 113 | 41 | 19 | 100 | 82 | 11 | 11 | 17 | 31 | 91 | 99 | 100 |
| 124 | 116 | 109 | 99 | 91 | 113 | 99 | 140 | 144 | 57 | 20 | 20 | 15 | 11 | 15 | 17 | 63 | 87 | 119 | 124 |
| 136 | 133 | 133 | 135 | 138 | 133 | 132 | 144 | 150 | 120 | 24 | 17 | 15 | 15 | 17 | 20 | 115 | 113 | 88 | 150 |
| 158 | 157 | 157 | 154 | 149 | 145 | 133 | 127 | 146 | 150 | 116 | 35 | 20 | 19 | 28 | 105 | 124 | 128 | 141 | 171 |
| 155 | 154 | 156 | 155 | 146 | 155 | 154 | 154 | 147 | 139 | 148 | 150 | 138 | 120 | 128 | 129 | 130 | 151 | 156 | 165 |
| 150 | 151 | 154 | 162 | 166 | 167 | 169 | 174 | 172 | 167 | 177 | 166 | 164 | 140 | 134 | 120 | 121 | 120 | 127 | 172 |
| 145 | 149 | 151 | 157 | 165 | 169 | 173 | 179 | 176 | 166 | 166 | 157 | 145 | 136 | 129 | 124 | 120 | 136 | 163 | 168 |
| 144 | 148 | 153 | 160 | 159 | 158 | 165 | 172 | 165 | 169 | 157 | 151 | 149 | 141 | 130 | 140 | 151 | 162 | 169 | 167 |
| 144 | 141 | 147 | 155 | 154 | 149 | 156 | 151 | 157 | 157 | 151 | 144 | 147 | 147 | 149 | 159 | 158 | 159 | 166 | 165 |
| 139 | 140 | 140 | 150 | 153 | 151 | 150 | 146 | 140 | 139 | 138 | 140 | 145 | 151 | 149 | 156 | 156 | 162 | 162 | 161 |
| 136 | 134 | 138 | 146 | 156 | 164 | 153 | 146 | 145 | 136 | 139 | 139 | 140 | 141 | 149 | 157 | 159 | 161 | 169 | 166 |
| 136 | 133 | 136 | 135 | 144 | 159 | 168 | 159 | 151 | 142 | 141 | 145 | 139 | 146 | 153 | 156 | 164 | 167 | 172 | 168 |
| 133 | 129 | 140 | 142 | 146 | 159 | 167 | 165 | 154 | 151 | 146 | 141 | 147 | 154 | 156 | 160 | 161 | 157 | 153 | 154 |

# Color images

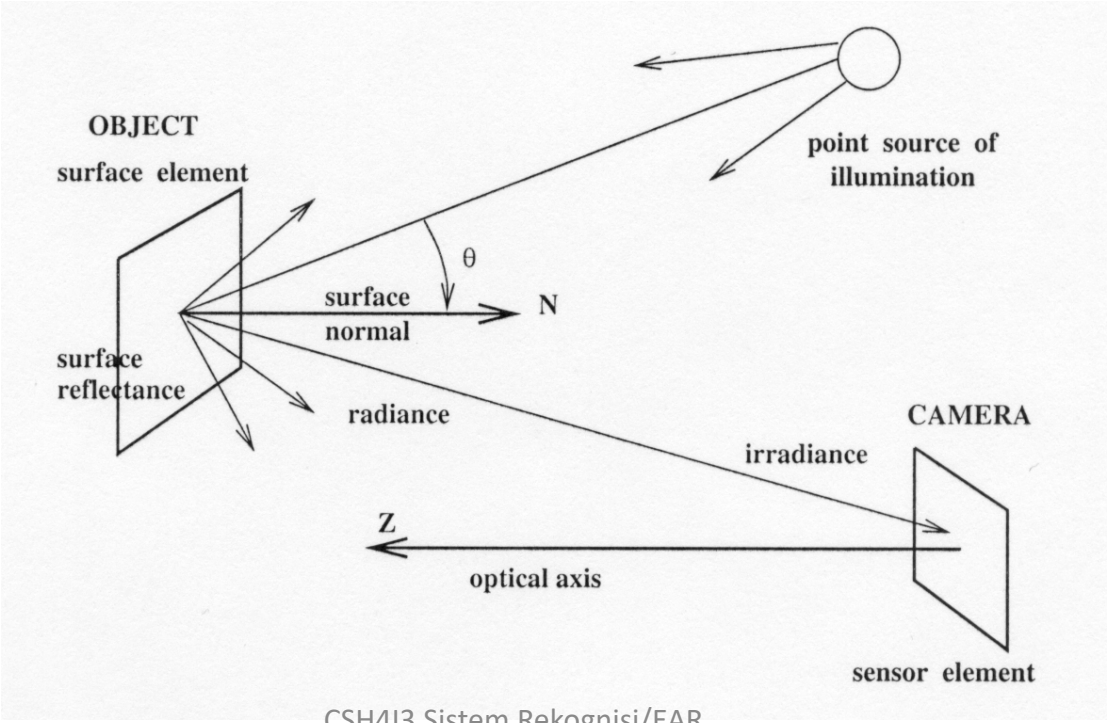


# Image formation

- There are two parts to the image formation process:
  - The **geometry of image formation**, which determines where in the image plane the projection of a point in the scene will be located.
  - The **physics of light**, which determines the brightness of a point in the image plane as a function of illumination and surface properties.

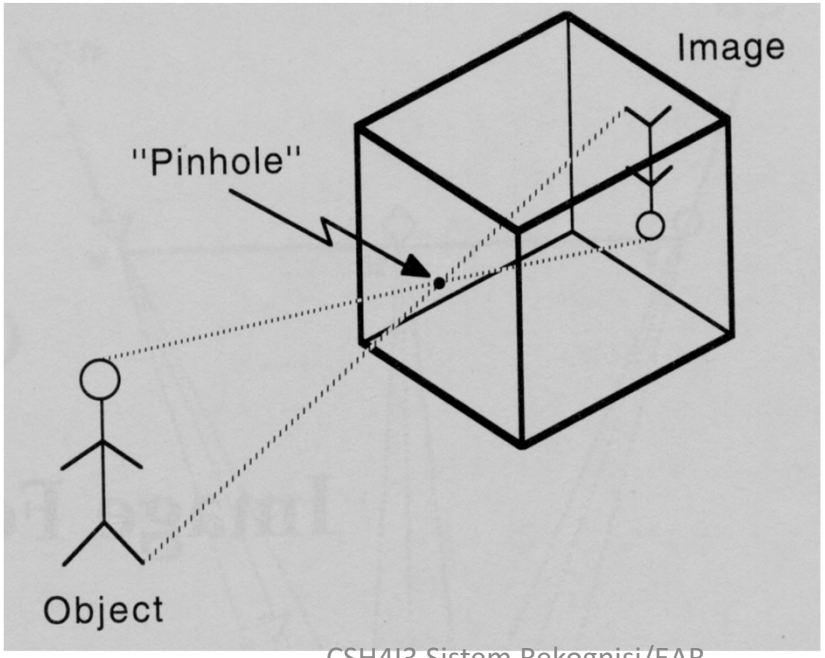
# A Simple model of image formation

- The scene is illuminated by a single source.
- The scene reflects radiation towards the camera.
- The camera senses it via chemicals on film.



# Pinhole camera

- This is the simplest device to form an image of a 3D scene on a 2D surface.
- Straight rays of light pass through a “pinhole” and form an inverted image of the object on the image plane.

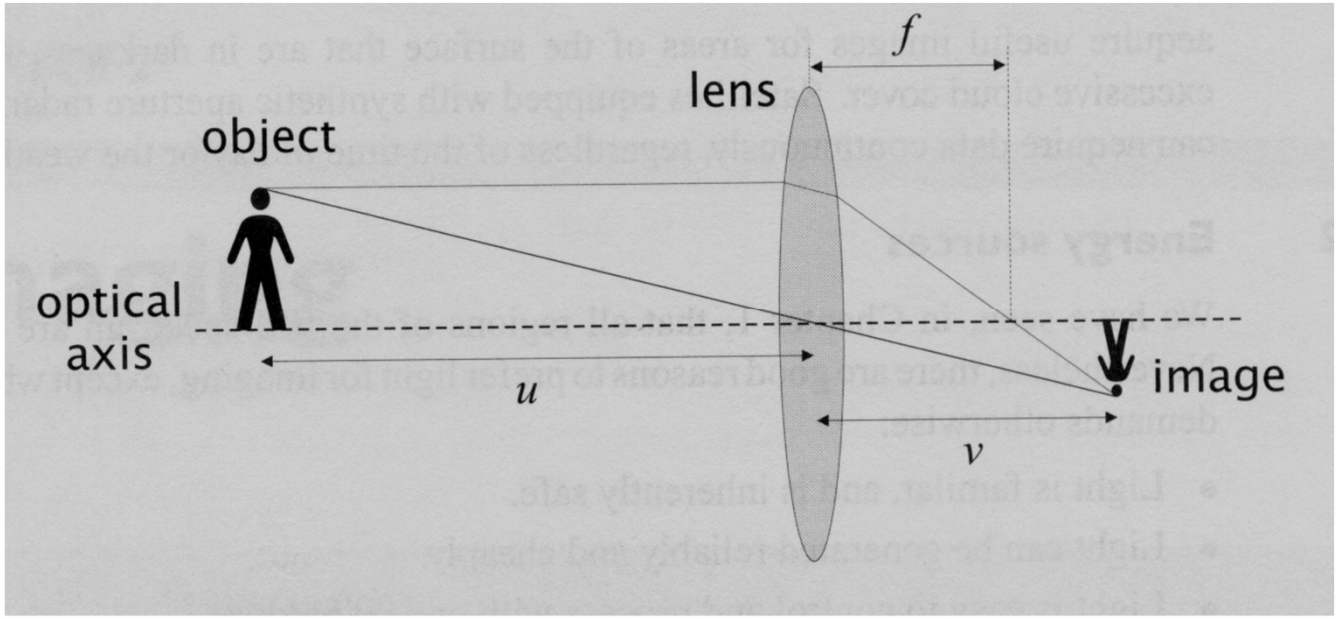



$$x = \frac{fX}{Z}$$

$$y = \frac{fY}{Z}$$

# Camera optics

- In practice, the aperture must be larger to admit more light.
- Lenses are placed to in the aperture to **focus** the bundle of rays from each scene point onto the corresponding point in the image plane



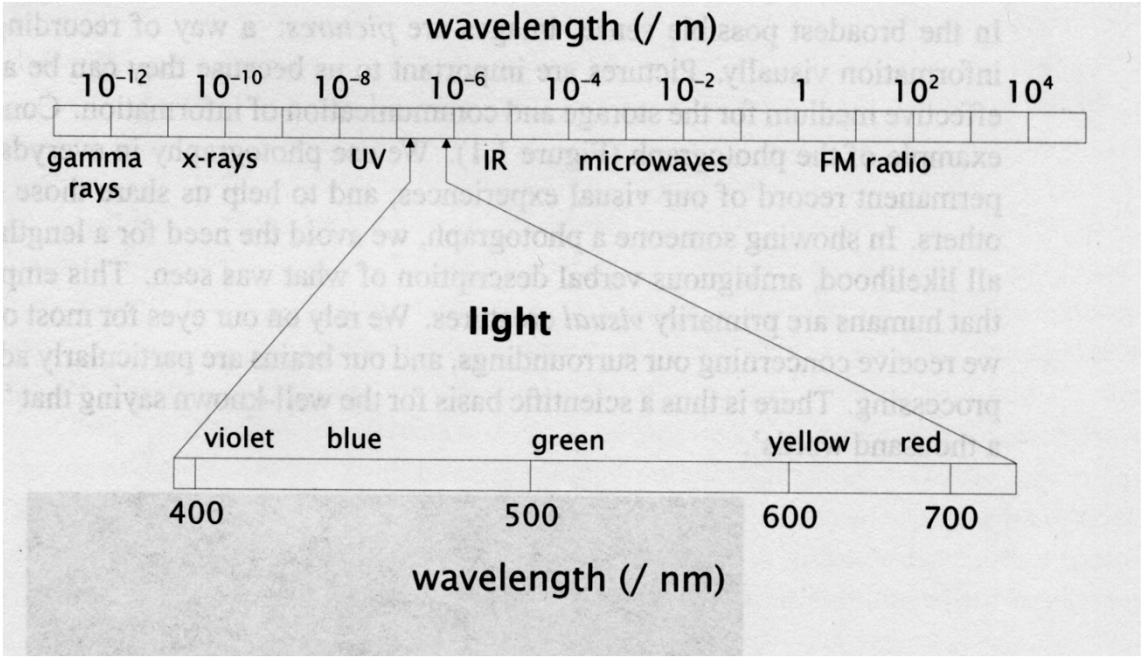
# Image formation (cont'd)

- Optical parameters of the lens
  - lens type
  - focal length
  - field of view
- Photometric parameters
  - type, intensity, and direction of illumination
  - reflectance properties of the viewed surfaces
- Geometric parameters
  - type of projections
  - position and orientation of camera in space
  - perspective distortions introduced by the imaging process

# Image distortion

# What is light?

- The visible portion of the **electromagnetic** (EM) spectrum.
- It occurs between wavelengths of approximately 400 and 700 nanometers.



# Short wavelengths

- Different wavelengths of radiation have different properties.
- The **x-ray** region of the spectrum, it carries sufficient energy to penetrate a significant volume or material.

# Long wavelengths

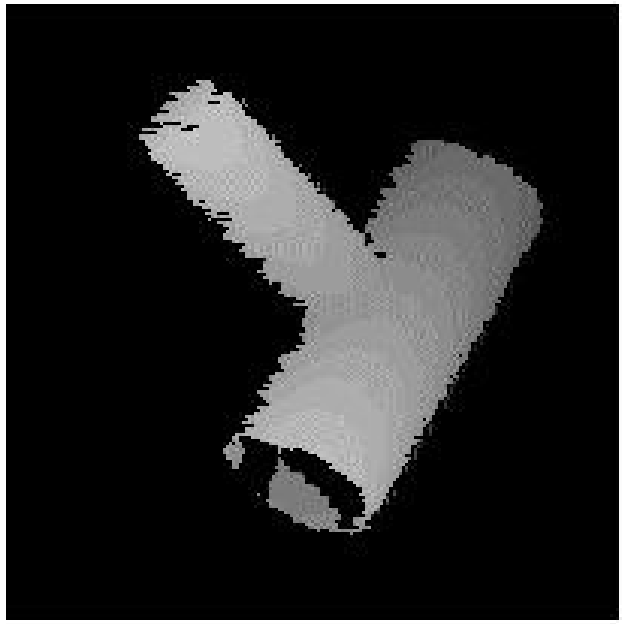
- Copious quantities of **infrared** (IR) radiation are emitted from warm objects (e.g., locate people in total darkness).

# Long wavelengths (cont'd)

- "**Synthetic aperture radar**" (SAR) imaging techniques use an artificially generated source of microwaves to probe a scene.
- SAR is unaffected by weather conditions and clouds (e.g., has provided us images of the surface of Venus).

# Range images

- An array of distances to the objects in the scene.
- They can be produced by sonar or by using laser rangefinders.

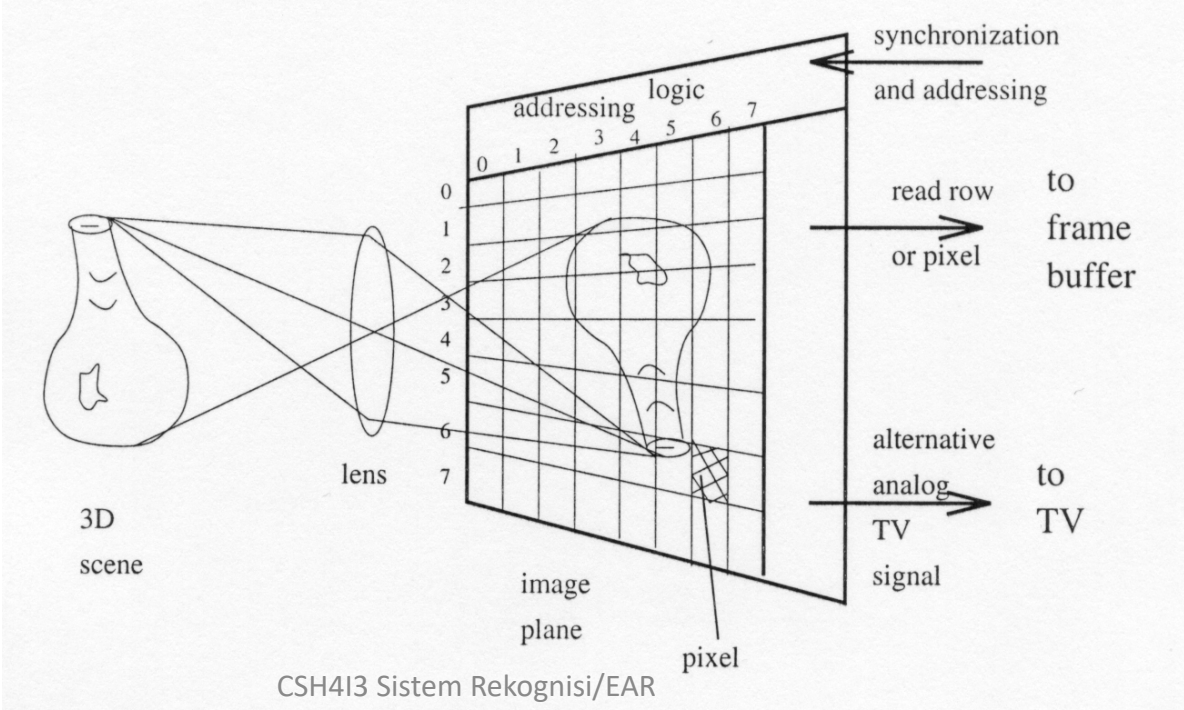


# Sonic images

- Produced by the reflection of sound waves off an object.
- High sound frequencies are used to improve resolution.

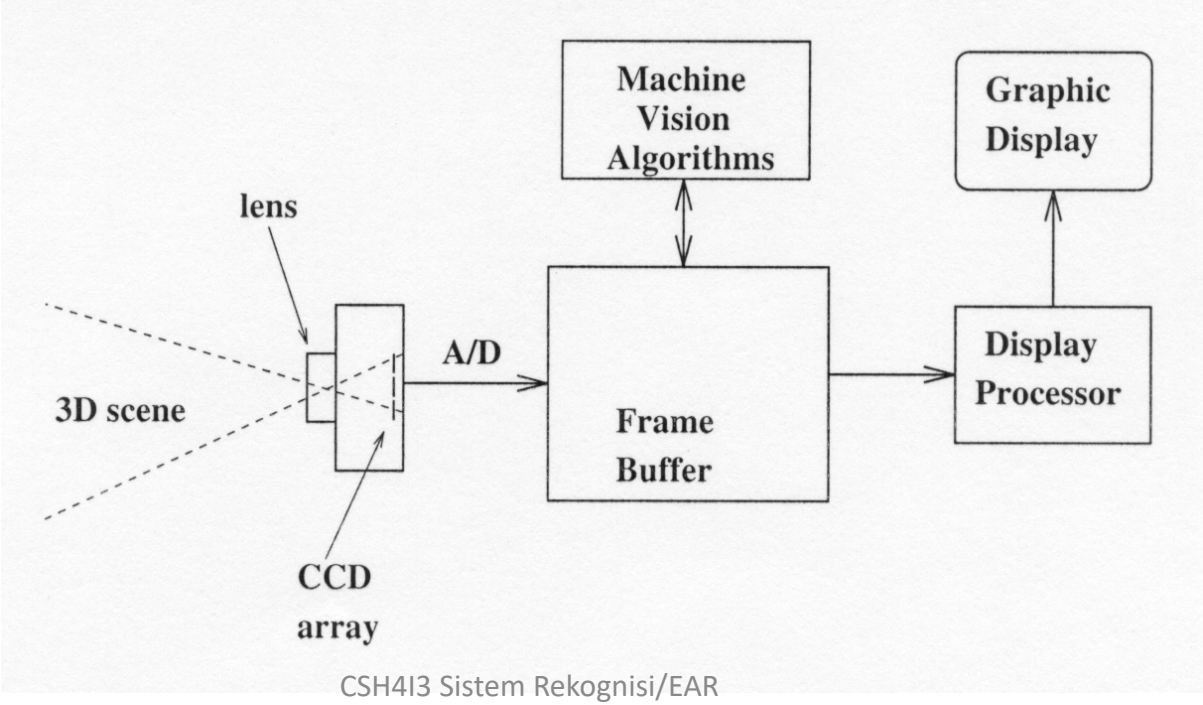
# CCD (Charged-Coupled Device) cameras

- Tiny solid state cells convert light energy into electrical charge.
- The image plane acts as a digital memory that can be read row by row by a computer.

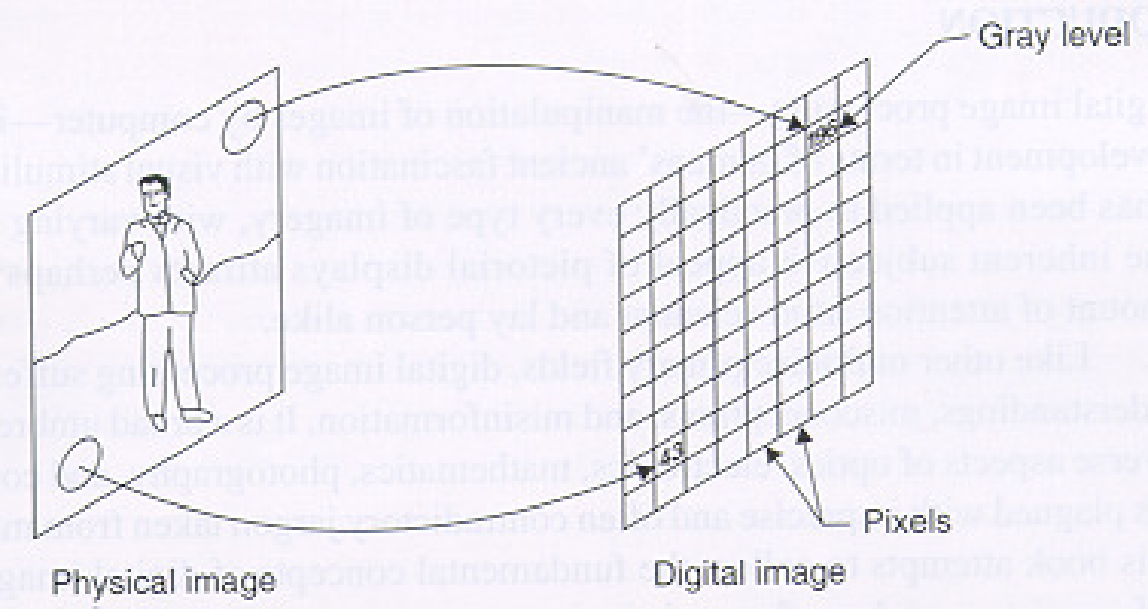


# Frame grabber

- Usually, a CCD camera plugs into a computer board (**frame grabber)**.
- The frame grabber digitizes the signal and stores it in its memory (**frame buffer**).

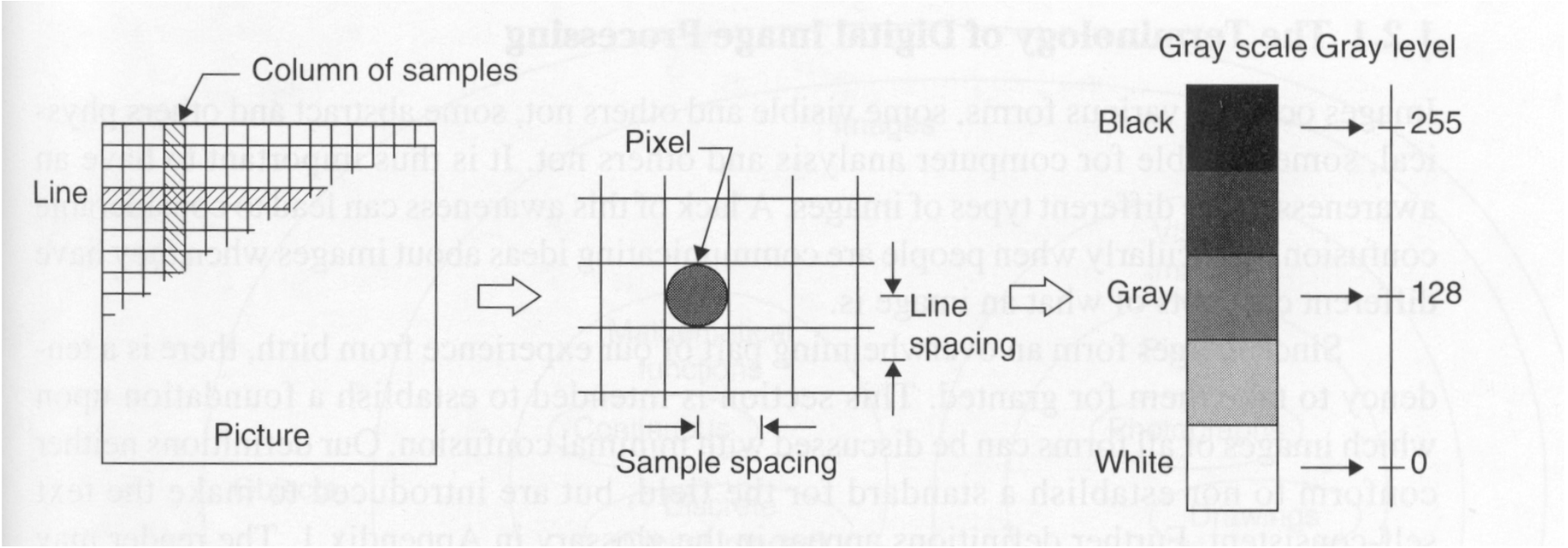


# Image digitization




- **Sampling** means measuring the value of an image at a finite number of points.
- **Quantization** is the representation of the measured value at the sampled point by an integer.

# Image digitization (cont'd)



# Image quantization(example)

- 256 gray levels (8bits/pixel) 32 gray levels (5 bits/pixel) 16 gray levels (4 bits/pixel)



- 8 gray levels (3 bits/pixel) 4 gray levels (2 bits/pixel) 2 gray levels (1 bit/pixel)



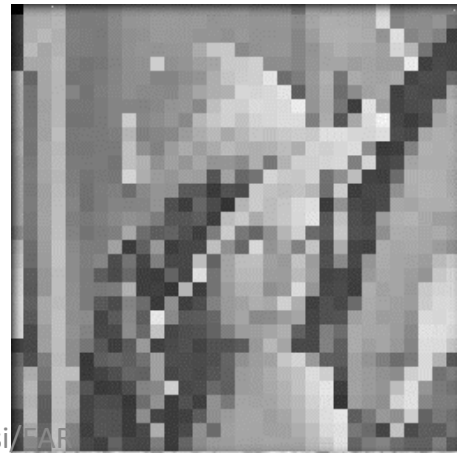
# Image sampling (example)

original image

sampled by a factor of 2

sampled by a factor of 4

sampled by a factor of 8



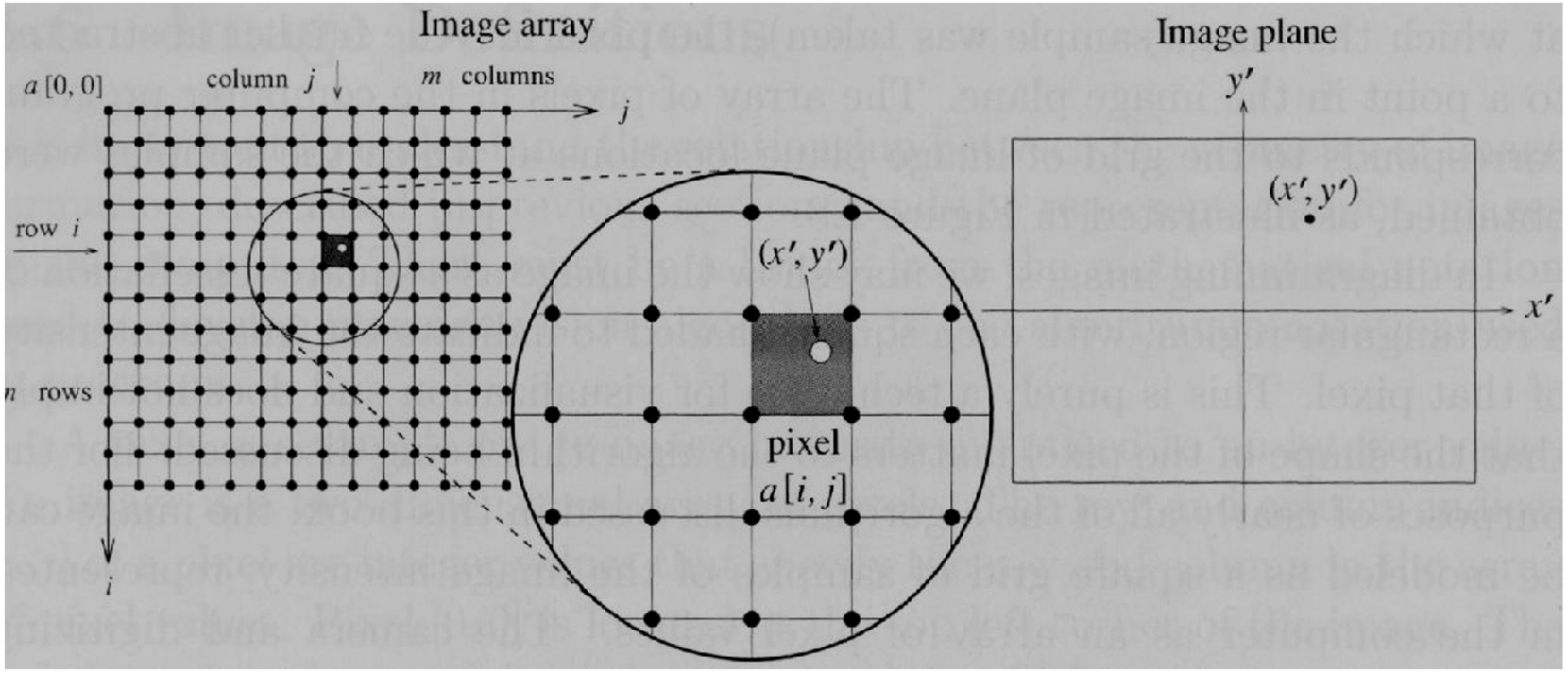
# Digital image

- An image is represented by a rectangular array of integers.
- An integer represents the brightness or darkness of the image at that point.
- N: # of rows, M: # of columns, Q: # of gray levels
  - N = $2^n$, M = $2^m$, Q = $2^q$ (q is the # of bits/pixel)
  - Storage requirements: NxMxQ (e.g., N=M=1024, q=8, 1MB)

$$\begin{matrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \dots & f(1,M-1) \\ \dots & \dots & \dots & \dots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{matrix}$$

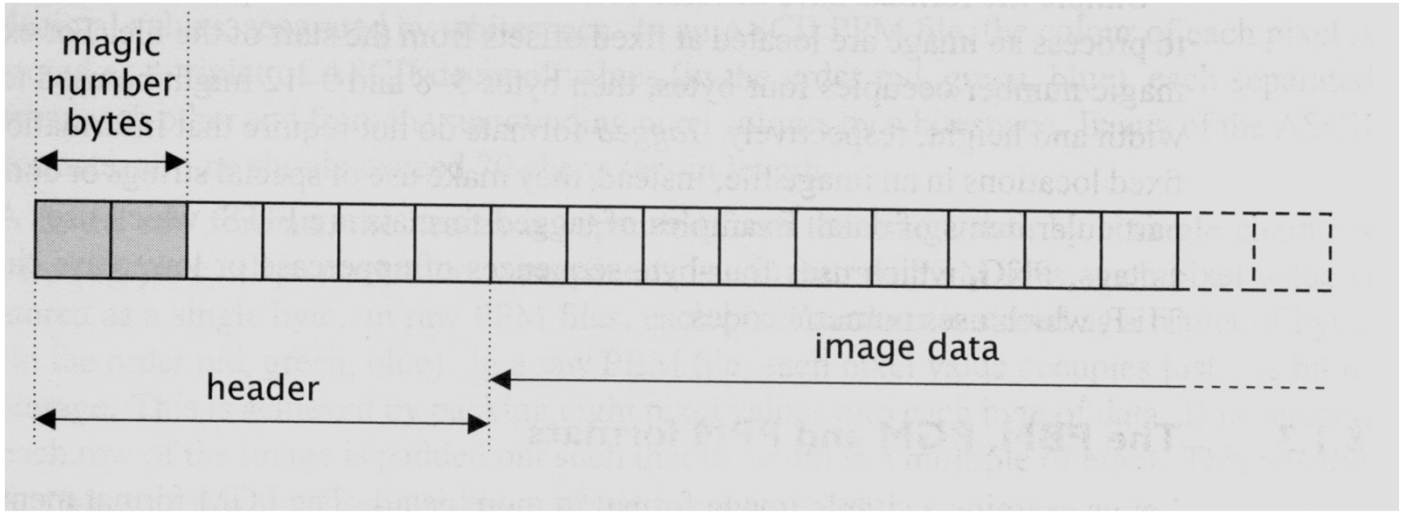
# Image coordinate system

$$x = j - \frac{N-1}{2} \qquad y = -(i - \frac{M-1}{2})$$



# Image file formats

- Many image formats adhere to the simple model shown below (line by line, no breaks between lines).
- The header contains at least the width and height of the image.
- Most headers begin with a **signature** or “magic number” - a short sequence of bytes for identifying the file format.



# Common image file formats

- GIF (Graphic Interchange Format) -
- PNG (Portable Network Graphics)
- JPEG (Joint Photographic Experts Group)
- TIFF (Tagged Image File Format)
- PGM (Portable Gray Map)
- FITS (Flexible Image Transport System)

# Comparison of image formats

| Image File Format | No. Bytes "Hi" | No. Bytes "Cars" |
|---|---|---|
| PGM | 595 | 509,123 |
| GIF | 192 | 138,267 |
| TIF | 918 | 171,430 |
| PS | 1591 | 345,387 |
| HIPS | 700 | 160,783 |
| JPG (lossless) | 684 | 49,160 |
| JPG (lossy) | 619 | 29,500 |

# PGM format

- A popular format for grayscale images (8 bits/pixel)
- Closely-related formats are:
  - PBM (Portable Bitmap), for binary images (1 bit/pixel)
  - PPM (Portable Pixelmap), for color images (24 bits/pixel)

```
P2
# a simple PGM image
7 7 255
 120 120 120 120 120 120 120
 120 120 120  33 120 120 120
 120 120 120  33 120 120 120
 120  33  33  33  33  33 120
 120 120 120  33 120 120 120
 120 120 120  33 120 120 120
 120 120 120 120 120 120 120
```

```
P5
# a simple PGM image
7 7 255
xxxxxxxxxx!xxxxxx!xxxx!!!!!xxxx!xxxxxx!xxxxxxxxx
```

ASCII or binary (raw) storage

Signatures of the various PBM, PGM and PPM image formats.

| Signature | Image type | Storage type |
|---|---|---|
| P1 | binary | ASCII |
| P2 | greyscale | ASCII |
| P3 | RGB | ASCII |
| P4 | binary | raw bytes |
| P5 | greyscale | raw bytes |
| P6 | RGB | raw bytes |

# ASCII vs Raw format

- ASCII format has the following advantages:
  - Pixel values can be examined or modified very easily using a standard text editor.
  - Files in raw format cannot be modified in this way since they contain many unprintable characters.
- Raw format has the following advantages:
  - It is much more compact compared to the ASCII format.
  - Pixel values are coded using only a single character !

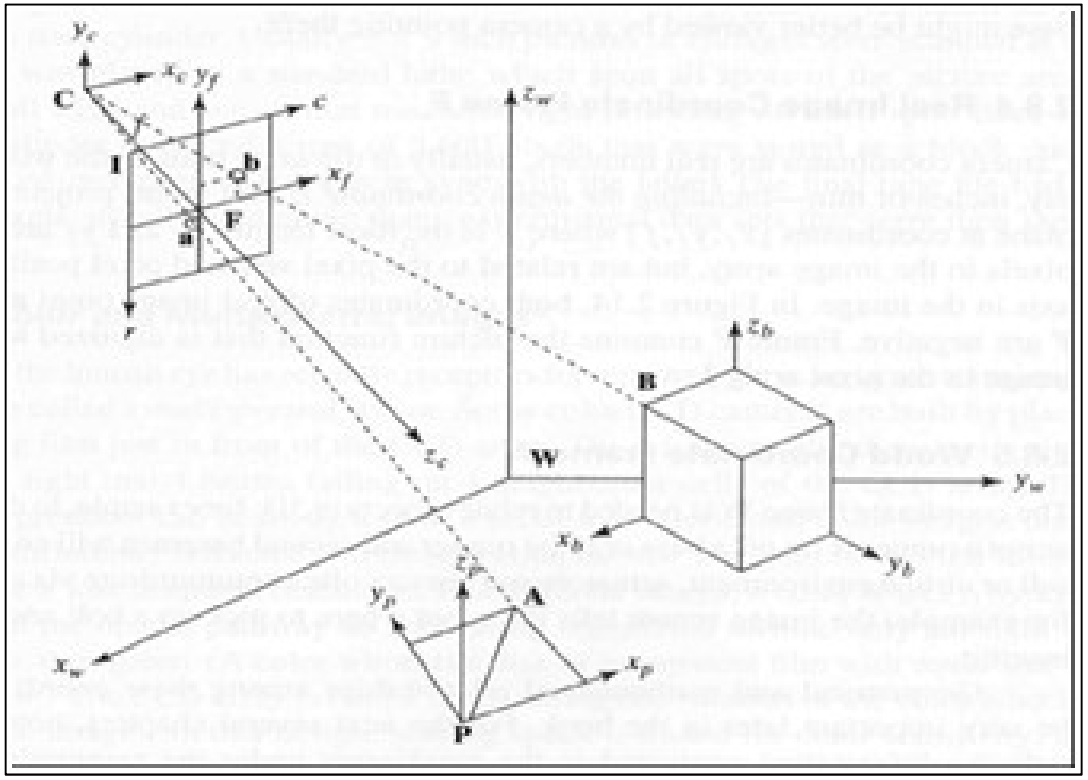
# Digital Image Processing

# Perspective Transformation

- When human eyes see near things they look bigger as compare to those who are far away. This is called perspective in a general way. Whereas transformation is the transfer of an object etc from one state to another.
- So overall , the perspective transformation deals with the conversion of 3d world into 2d image. The same principle on which human vision works and the same principle on which the camera works.

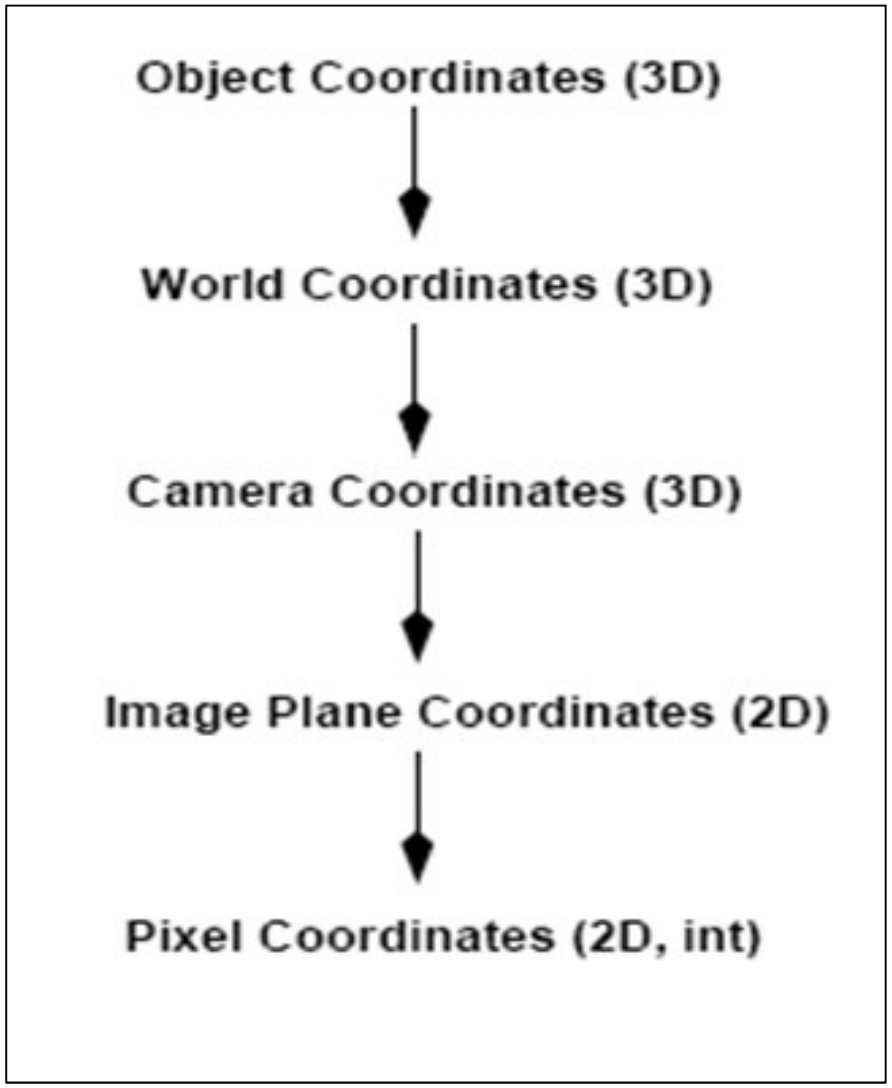
# Frame of Reference

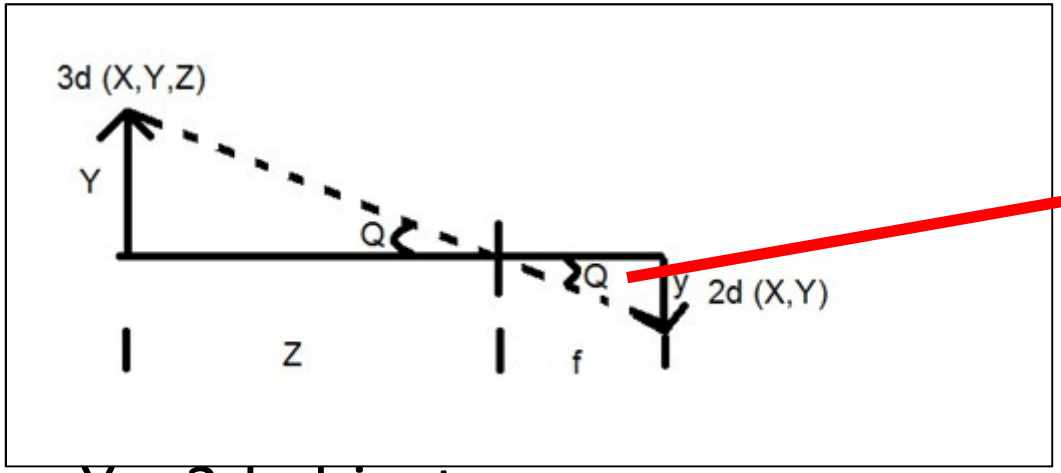
- a set of values in relation to which we measure something.



# 5 FRAMES OF REFERENCE

- Object
  - OBJECT COORDINATE FRAME:Object coordinate frame is used for modeling objects. For example , checking if a particular object is in a proper place with respect to the other object. It is a 3d coordinate system.
- World
  - WORLD COORDINATE FRAME: World coordinate frame is used for co-relating objects in a 3 dimensional world. It is a 3d coordinate system.
- Camera
  - CAMERA COORDINATE FRAME: Camera co-ordinate frame is used to relate objects with respect of the camera. It is a 3d coordinate system.
- Image
  - IMAGE COORDINATE FRAME: It is not a 3d coordinate system , rather it is a 2d system. It is used to describe how 3d points are mapped in a 2d image plane.
- Pixel
  - PIXEL COORDINATE FRAME: It is also a 2d coordinate system. Each pixel has a value of pixel co ordinates.


Object Coordinates (3D)
World Coordinates (3D)
Camera Coordinates (3D)
Image Plane Coordinates (2D)
Pixel Coordinates (2D, int)




$$\tan\theta = -\frac{y}{f}$$

$$\tan\theta = \frac{Y}{Z}$$

- Y = 3d object
- y = 2d Image
- f = focal length of the camera
- Z = distance between image and the camera

$$Y = -f\frac{Y}{Z}$$

- From this equation, we can see that when the rays of light reflect back after striking from the object , passed from the camera , an invert image is formed.

# Calculating the size of image formed

- Suppose an image has been taken of a person 5m tall, and standing at a distance of 50m from the camera, and we have to tell that what is the size of the image of the person, with a camera of focal length is 50mm.
- SOLUTION:

  Since the focal length is in millimeter , so we have to convert every thing in millimeter in order to calculate it. So,

Y = 5000 mm.

f = 50 mm.

Z = 50000 mm.

$$Y = -f\frac{Y}{Z} = -50 \times 5000 / 50000$$

- Putting the values in the formula , we get = -5 mm. Again, the minus sign indicates that the image is inverted.

| Bits per pixel | Number of colors |
|---|---|
| 1 bpp | 2 colors |
| 2 bpp | 4 colors |
| 3 bpp | 8 colors |
| 4 bpp | 16 colors |
| 5 bpp | 32 colors |
| 6 bpp | 64 colors |
| 7 bpp | 128 colors |
| 8 bpp | 256 colors |
| 10 bpp | 1024 colors |
| 16 bpp | 65536 colors |
| 24 bpp | 16777216 colors (16.7 million colors) |
| 32 bpp | 4294967296 colors (4294 million colors) |

$$\text{White color} = (2)^{bpp} - 1$$

- In case of 1 bpp , 0 denotes black , and 1 denotes white.
- In case 8 bpp , 0 denotes black , and 255 denotes white.
- *) bpp=bit per pixel

# IMAGE SIZE

- The size of an image depends upon three things.
  - Number of rows
  - Number of columns
  - Number of bits per pixel
- The formula for calculating the size = rows * cols * bpp
- Assuming an image has 1024 rows and it has 1024 columns. And since it is a gray scale image , it has 256 different shades of gray or it has bits per pixel. Then putting these values in the formula , we get size of an image = rows * cols * bpp

  = 1024 * 1024 * 8

  = 8388608 bits.
- But since its not a standard answer that we recognize , so will convert it into our format.
  - Converting it into bytes = 8388608 / 8 = 1048576 bytes.
  - Converting into kilo bytes = 1048576 / 1024 = 1024kb.
  - Converting into Mega bytes = 1024 / 1024 = 1 Mb.

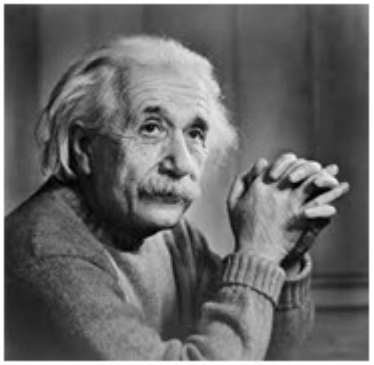
# Types of Images

- The binary image
  - The binary image as it name states , contain only two pixel values, 0 and 1.
  - The resulting image that is formed hence consist of only black and white color and thus can also be called as Black and White image.
  - NO GRAY LEVEL
  - One of the interesting this about this binary image that there is no gray level in it. Only two colors that are black and white are found in it.
  - FORMAT: Binary images have a format of PBM (Portable bit map )

# 2 , 3 , 4 ,5 ,6 bit color format

- The images with a color format of 2 , 3 , 4 ,5 and 6 bit are not widely used today. They were used in old times for old TV displays , or monitor displays.
- But each of these colors have more then two gray levels , and hence has gray color unlike the binary image.
- In a 2 bit 4, in a 3 bit 8 , in a 4 bit 16, in a 5 bit 32, in a 6 bit 64 different colors are present.

# 8 bit color format

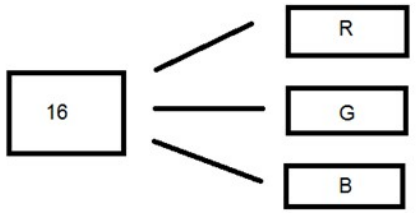


- 8 bit color format is one of the most famous image format. It has 256 different shades of colors in it. It is commonly known as Grayscale image.
- The range of the colors in 8 bit vary from 0-255. Where 0 stands for black , and 255 stands for white , and 127 stands for gray color.
- This format was used initially by early models of the operating systems UNIX and the early color Macintoshes.
- FORMAT: The format of these images are PGM ( Portable Gray Map ).

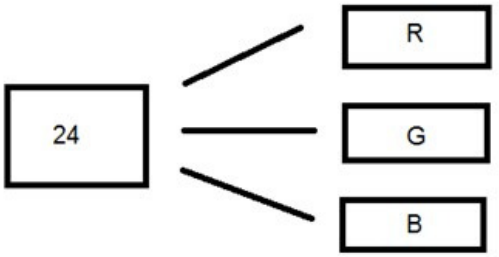
# 16 bit color format

- It is a color image format. It has 65,536 different colors in it. It is also known as High color format.
- It has been used by Microsoft in their systems that support more then 8 bit color format
- The distribution of color in a color image is not as simple as it was in grayscale image.
- A 16 bit format is actually divided into three further formats which are Red , Green and Blue. The famous (RGB) format.

- Now the question arises , that how would you distribute 16 into three. If you do it like this, 5 bits for R , 5 bits for G , 5 bits for B
  - Then there is one bit remains in the end.
- So the distribution of 16 bit has been done like this. 5 bits for R , 6 bits for G , 5 bits for B.
  - The additional bit that was left behind is added into the green bit. Because green is the color which is most soothing to eyes in all of these three colors.
  - Note this is distribution is not followed by all the systems. Some have introduced an alpha channel in the 16 bit.
- ANOTHER DISTRIBUTION OF 16 BIT FORMAT IS LIKE THIS: 4 bits for R , 4 bits for G , 4 bits for B , 4 bits for alpha channel.
- Or some distribute it like this 5 bits for R , 5 bits for G , 5 bits for B , 1 bits for alpha channel.



# 24 bit color format




- Since 24 is equally divided on 8 , so it has been distributed equally between three different color channels. Their distribution is like this. 8 bits for R , 8 bits for G , 8 bits for B.
- Unlike a 8 bit gray scale image , which has one matrix behind it, a 24 bit image has three different matrices of R , G , B.
- FORMAT
  - It is the most common used format. Its format is PPM ( Portable pixMap) which is supported by Linux operating system. The famous windows has its own format for it which is BMP ( Bitmap ).

# Common colors and their Hex code

| Color | Hex Code |
|---|---|
| Black | #000000 |
| White | #FFFFFF |
| Gray | #808080 |
| Red | #FF0000 |
| Green | #00FF00 |
| Blue | #0000FF |
| Cyan | #00FFFF |
| Magenta | #FF00FF |
| Yellow | #FFFF00 |

# Pixel Resolution

- Pixel:
  - smallest element of an image; a pixel can store a value proportional to the light intensity at that particular location.
- Pixel Resolution
  - total number of count of pixels in an digital image
  - can be defined with set of two numbers. The first number the width of the picture , or the pixels across columns , and the second number is height of the picture , or the pixels across its width.
  - the higher is the pixel resolution , the higher is the quality of the image.

- Megapixels
  - Column pixels (width ) X row pixels ( height ) / 1 Million.
  - Lets say we have an image of dimension: 2500 X 3192. Its pixel resolution = 2500 * 3192 = 7982350 bytes. Dividing it by 1 million = 7.9 = 8 mega pixel (approximately).

- The size of an image can be defined by its pixel resolution. Size = pixel resolution X bpp ( bits per pixel )

# Aspect ratio

- the ratio between width of an image and the height of an image. It is commonly explained as two numbers separated by a colon (8:9). This ratio differs in different images , and in different screens. The common aspect ratios are: 1.33:1, 1.37:1, 1.43:1, 1.50:1, 1.56:1, 1.66:1, 1.75:1, 1.78:1, 1.85:1, 2.00:1, e.t.c
- With the aspect ratio, you can calculate the dimensions of the image along with the size of the image.
- ADVANTAGE:
  - Aspect ratio maintains a balance between the appearance of an image on the screen , means it maintains a ratio between horizontal and vertical pixels. It does not let the image to get distorted when aspect ratio is increased.

- If you are given an image with aspect ratio of 6:2 of an image of pixel resolution of 480000 pixels given the image is an gray scale image. And you are asked to calculate two things.
  - Resolve pixel resolution to calculate the dimensions of image
  - Calculate the size of the image
- SOLUTION:
- GIVEN:
  - Aspect ratio: c:r = 6:2
  - Pixel resolution: c * r = 480000
  - Bits per pixel: grayscale image = 8bpp
- FIND:
  - Number of rows = ?
  - Number of cols = ?
- SOLVING FIRST PART:

Equation 1. c:r = 6:2 → c = 6 r / 2

Equation 2. c = 480000/r

Comparing both equations → $\frac{6r}{2} = \frac{480000}{r}$

$r^2 = \sqrt{\frac{480000*2}{6}}$

That gives r = 400.

Put r in equation 1, we get → c = 1200.

So rows = 400 cols = 1200.

- SOLVING 2ND PART:

  Size = rows * cols * bpp

  Size of image in bits = 400 * 1200 * 8 = 3840000 bits

  Size of image in bytes = 480000 bytes

  Size of image in kilo bytes = 48 kb (approx).

# Convolution

- It can be mathematically represented as two ways
  - **g(x,y) = h(x,y) * f(x,y)**
    - It can be explained as the “mask convolved with an image”.
  - Or
  - **g(x,y) = f(x,y) * h(x,y)**
    - It can be explained as “image convolved with mask”.
  - the convolution operator(*) is commutative. The h(x,y) is the mask or filter.
- Mask is also a signal. It can be represented by a two dimensional matrix. The mask is usually of the order of 1x1, 3x3, 5x5 , 7x7 . A mask should always be in odd number , because other wise you cannot find the mid of the mask.

# HOW TO PERFORM CONVOLUTION?

- Flip the mask (horizontally and vertically) only once
- Slide the mask onto the image.
- Multiply the corresponding elements and then add them
- Repeat this procedure until all values of the image has been calculated.

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

Mask

| | | |
|---|---|---|
| 3 | 2 | 1 |
| 6 | 5 | 4 |
| 9 | 8 | 7 |

Flip horisontally

| | | |
|---|---|---|
| 9 | 8 | 7 |
| 6 | 5 | 4 |
| 3 | 2 | 1 |

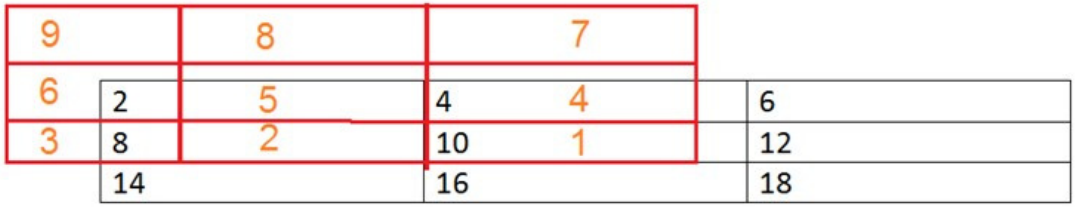
Flip Vertically

| | | |
|---|---|---|
| 2 | 4 | 6 |
| 8 | 10 | 12 |
| 14 | 16 | 18 |

Image

# Convolution

- Place the center of the mask at each element of an image. Multiply the corresponding elements and then add them , and paste the result onto the element of the image on which you place the center of mask.




- The box in red color is the mask , and the values in the orange are the values of the mask. The black color box and values belong to the image. Now for the first pixel of the image , the value will be calculated as

  First pixel = (5*2) + (4*4) + (2*8) + (1*10)

  = 10 + 16 + 16 + 10 = 52

  Place 52 in the original image at the first index and repeat this procedure for each pixel of the image.

# Mask

- A mask is a filter. Concept of masking is also known as spatial filtering. Masking is also known as filtering. In this concept we just deal with the filtering operation that is performed directly on the image.
- WHAT IS FILTERING.
  - The process of filtering is also known as convolving a mask with an image. As this process is same of convolution so filter masks are also known as convolution masks.
- HOW IT IS DONE.
  - The general process of filtering and applying masks is consists of moving the filter mask from point to point in an image. At each point (x,y) of the original image, the response of a filter is calculated by a pre defined relationship. All the filters values are pre defined and are a standard.
- TYPES OF FILTERS
  - Generally there are two types of filters. One is called as linear filters or smoothing filters and others are called as frequency domain filters.
- WHY FILTERS ARE USED?
  - Filters are applied on image for multiple purposes. The two most common uses are as following:
    - Filters are used for Blurring and noise reduction
    - Filters are used or edge detection and sharpness

- BLURRING AND NOISE REDUCTION:
  - Filters are most commonly used for blurring and for noise reduction. Blurring is used in pre processing steps, such as removal of small details from an image prior to large object extraction.
- MASKS FOR BLURRING.
  - The common masks for blurring are.
    - Box filter
    - Weighted average filter
- In the process of blurring we reduce the edge content in an image and try to make the transitions between different pixel intensities as smooth as possible.
- Noise reduction is also possible with the help of blurring.

- EDGE DETECTION AND SHARPNESS:
  - Masks or filters can also be used for edge detection in an image and to increase sharpness of an image.
- WHAT ARE EDGES.
  - We can also say that sudden changes of discontinuities in an image are called as edges. Significant transitions in an image are called as edges

# Blurring

- in blurring , we simple reduce the edge content and makes the transition form one color to the other very smooth.
- Types of filters.
  - Mean filter → known as Box filter or average filter.
  - Weighted average filter
  - Gaussian filter

| | | |
|---|---|---|
| 1/9 | 1/9 | 1/9 |
| 1/9 | 1/9 | 1/9 |
| 1/9 | 1/9 | 1/9 |

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 10 | 1 |
| 1 | 1 | 1 |

# Edges

- sudden changes of discontinuities in an image are called as edges; Significant transitions in an image are called as edges.
- WHY DETECT EDGES.
  - Most of the shape information of an image is enclosed in edges. So first we detect these edges in an image by using these filters and then by enhancing those areas of image which contains edges, sharpness of the image will increase and image will become clearer.

# some of the masks for edge detection

- Prewitt Operator: detecting edges horizontally and vertically.
- Sobel Operator: very similar to Prewitt operator. It also calculates edges in both horizontal and vertical direction.
- Robinson Compass Masks: known as direction mask. In this operator we take one mask and rotate it in all the 8 compass major directions to calculate edges of each direction.
- Krisch Compass Masks: also a derivative mask which is used for finding edges. Kirsch mask is also used for calculating edges in all the directions.
- Laplacian Operator: also a derivative operator which is used to find edges in an image. Laplacian is a second order derivative mask. It can be further divided into positive laplacian and negative laplacian.

- All above → Linear filters or smoothing filters.

# Prewitt Operator

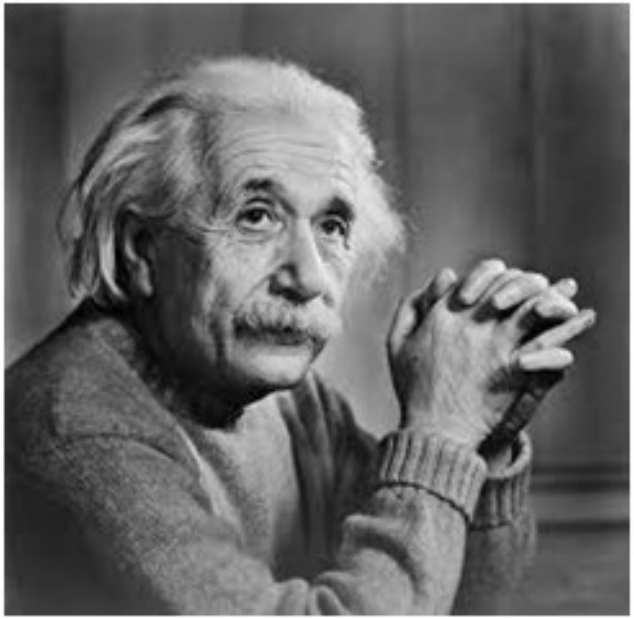
- Edges are calculated by using difference between corresponding pixel intensities of an image. All the masks that are used for edge detection are also known as derivative masks. image is also a signal so changes in a signal can only be calculated using differentiation. So that's why these operators are also called as derivative operators or derivative masks.
- All the derivative masks should have the following properties:
  - Opposite sign should be present in the mask.
  - Sum of mask should be equal to zero.
  - More weight means more edge detection.
- Prewitt operator provides us two masks one for detecting edges in horizontal direction and another for detecting edges in an vertical direction.

# Vertical Direction

| | | |
|---|---|---|
| -1 | 0 | 1 |
| -1 | 0 | 1 |
| -1 | 0 | 1 |

- When we apply this mask on the image it prominent vertical edges. It simply works like as first order derivate and calculates the difference of pixel intensities in a edge region. As the center column is of zero so it does not include the original values of an image but rather it calculates the difference of right and left pixel values around that edge. This increase the edge intensity and it become enhanced comparatively to the original image.

# Vertical mask

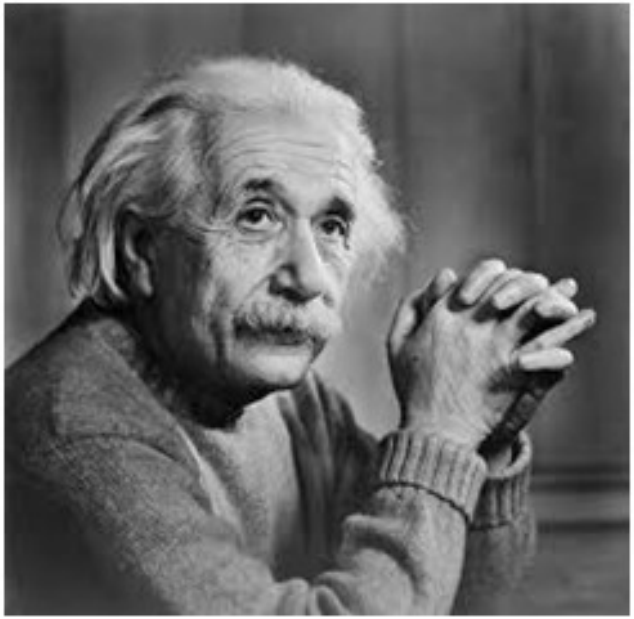


# Horisontal direction

| | | |
|---|---|---|
| -1 | -1 | -1 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |

- This mask will prominent the horizontal edges in an image. It also works on the principle of above mask and calculates difference among the pixel intensities of a particular edge. As the center row of mask is consist of zeros so it does not include the original values of edge in the image but rather it calculate the difference of above and below pixel intensities of the particular edge. Thus increasing the sudden change of intensities and making the edge more visible.
- Both the above masks follow the principle of derivate mask. Both masks have opposite sign in them and both masks sum equals to zero. The third condition will not be applicable in this operator as both the above masks are standardize and we can't change the value in them.

# Horisontal mask

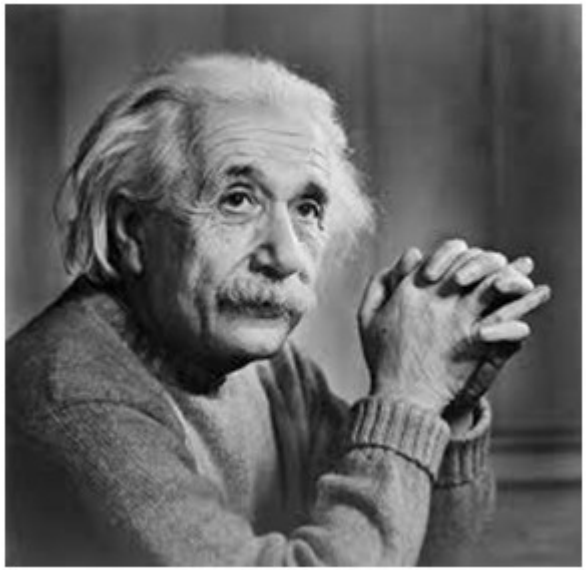


# Sobel Operator

- The major difference with Prewitt is that in sobel operator the coefficients of masks are not fixed and they can be adjusted according to our requirement unless they do not violate any property of derivative masks.

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

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



# Laplacian Operator

- Positive Laplacian Operator
  - Is use to take out outward edges in an image.

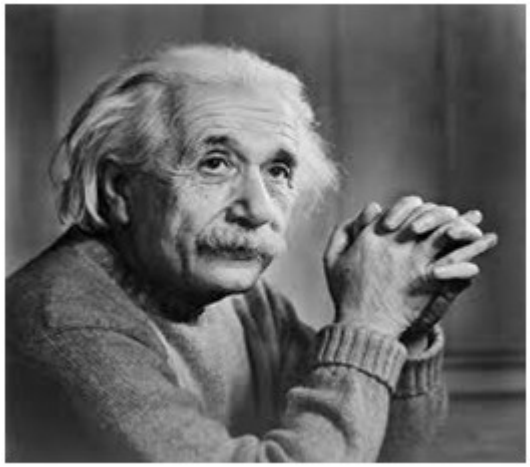
| | | |
|---|---|---|
| 0 | 1 | 0 |
| 1 | -4 | 1 |
| 0 | 1 | 0 |

- Negative Laplacian Operator
  - is use to take out inward edges in an image

| | | |
|---|---|---|
| 0 | -1 | 0 |
| -1 | 4 | -1 |
| 0 | -1 | 0 |

- Laplacian is a derivative operator; its uses highlight gray level discontinuities in an image and try to deemphasize regions with slowly varying gray levels. This operation in result produces such images which have grayish edge lines and other discontinuities on a dark background. This produces inward and outward edges in an image

- The important thing is how to apply these filters onto image. Remember we can't apply both the positive and negative Laplacian operator on the same image. we have to apply just one but the thing to remember is that if
  - we apply positive Laplacian operator on the image then we subtract the resultant image from the original image to get the sharpened image.
  - Similarly if we apply negative Laplacian operator then we have to add the resultant image onto original image to get the sharpened image.



positive

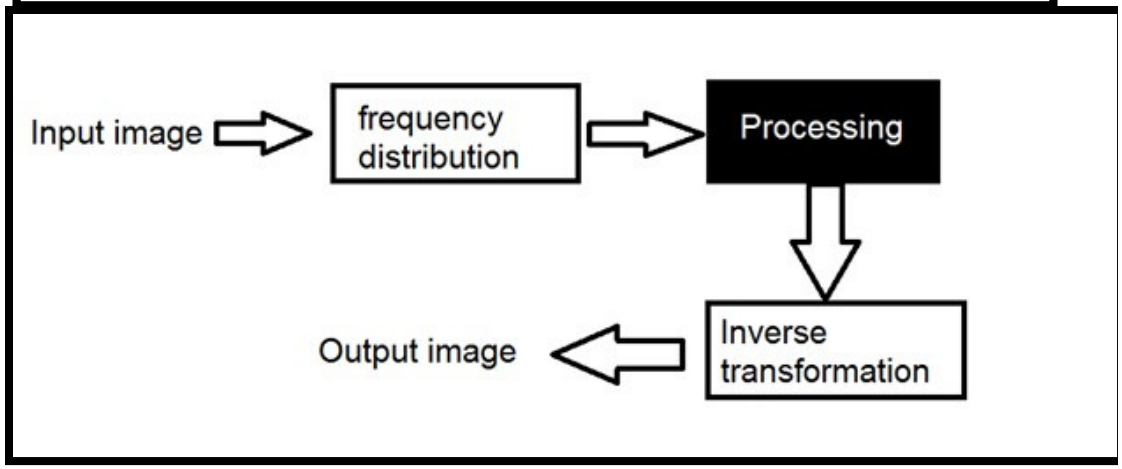
negative

# Frequency domain analysis

- Till now , all the domains in which we have analyzed a signal , we analyze it with respect to time. But in frequency domain we don’t analyze signal with respect to time , but with respect of frequency.
- DIFFERENCE BETWEEN SPATIAL DOMAIN AND FREQUENCY DOMAIN.
  - In spatial domain , we deal with images as it is. The value of the pixels of the image change with respect to scene. Whereas in frequency domain , we deal with the rate at which the pixel values are changing in spatial domain.

- Spatial domain


- Frequency domain



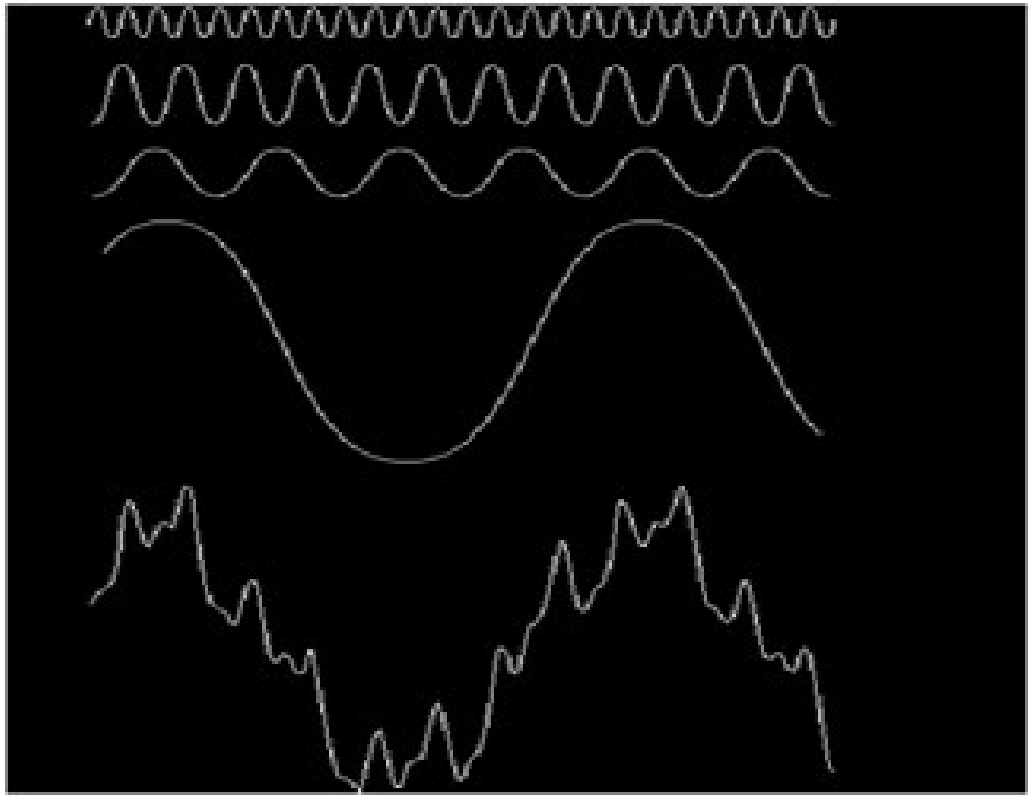

- TRANSFORMATION
  - A signal can be converted from time domain into frequency domain using mathematical operators called transforms
    - Fourier Series
    - Fourier transformation
    - Laplace transform
    - Z transform

# FREQUENCY COMPONENTS

- Any image in spatial domain can be represented in a frequency domain. But what do this frequencies actually mean.
- **HIGH FREQUENCY COMPONENTS**
  - High frequency components correspond to edges in an image.
- **LOW FREQUENCY COMPONENTS**
  - Low frequency components in an image correspond to smooth regions.

# Fourier Series and Transform

- Fourier was a mathematician in 1822. He give Fourier series and Fourier transform to convert a signal into frequency domain.
- Fourier series simply states that, periodic signals can be represented into sum of sines and cosines when multiplied with a certain weight.
  - It further states that periodic signals can be broken down into further signals with the following properties.
    - The signals are sines and cosines
    - The signals are harmonics of each other



- the last signal is actually the sum of all the above signals. This was the idea of the Fourier.

- in order to process an image in frequency domain , we need to first convert it using into frequency domain and we have to take inverse of the output to convert it back into spatial domain.
  - That's why both Fourier series and Fourier transform has two formulas. One for conversion and one converting it back to the spatial domain.

$$F(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(ux+vy)} \, dx \, d$$

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) e^{j2\pi(ux+vy)} \, du \, dv.$$

# Fourier Transform

- that the non periodic signals whose area under the curve is finite can also be represented into integrals of the sines and cosines after being multiplied by a certain weight. The Fourier transform has many wide applications that include , image compression (eg. JPEG compression) , filtering and image analysis.
- DIFFERENCE BETWEEN FOURIER SERIES AND TRANSFORM
  - Fourier series is applied on periodic signals and Fourier transform is applied for non periodic signals
- WHICH ONE IS APPLIED ON IMAGES.
  - the answer to this question lies in the fact that what images are non – periodic. so Fourier transform is used to convert them into frequency domain.
  - DISCRETE FOURIER TRANSFORM.
    - Since we are dealing with images, and in fact digital images , so for digital images we will be working on discrete fourier transform

- Consider the above Fourier term of a sinusoid. It include three things.
  - Spatial Frequency: relates with the brightness of the image
  - Magnitude:  relates with the contrast. Contrast is the difference between maximum and minimum pixel intensity.
  - Phase: contains the color information.

- 2D discrete Fourier Transform:

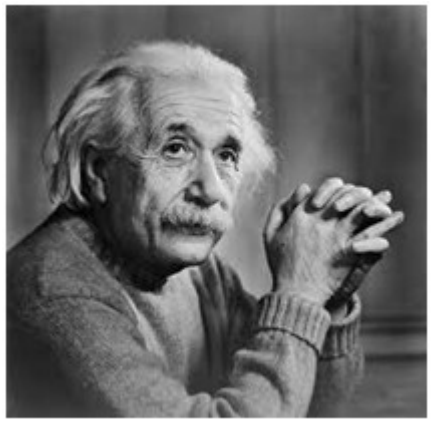
$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M+vy/N)}$$

- The discrete Fourier transform is actually the sampled Fourier transform, so it contains some samples that denotes an image. In the above formula f(x,y) denotes the image, and F(u,v) denotes the discrete Fourier transform. The formula for 2 dimensional inverse discrete Fourier transform is given below.

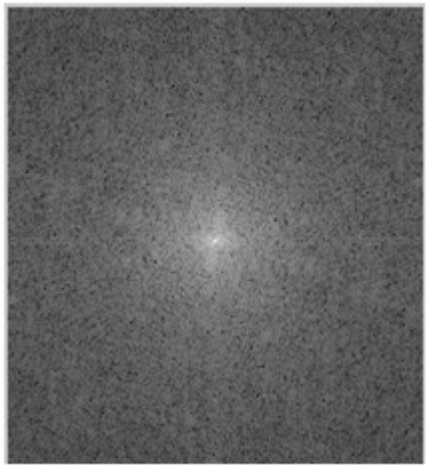
$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(ux/M+vy/N)}$$

# Convolution Theorem

Spatial domain



Frequency domain



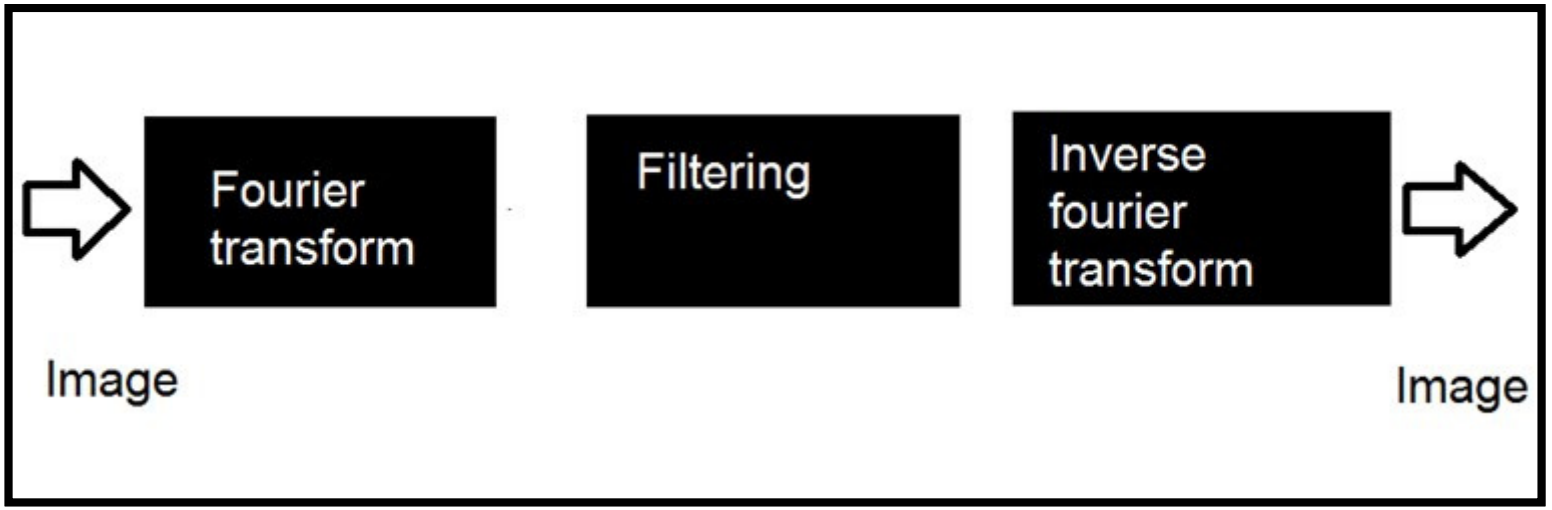
- relationship can be explained by a theorem which is called as Convolution theorem.

$$f(x,y) * h(x,y) \leftrightarrow F(u,v)H(u,v)$$

$$f(x,y)h(x,y) \leftrightarrow F(u,v) * H(u,v)$$

$$h(x,y) \leftrightarrow H(u,v)$$

- The filtering in frequency domain can be represented as following:



- **The steps in filtering**
  - At first step we have to do some pre – processing an image in spatial domain, means increase its contrast or brightness
  - Then we will take discrete Fourier transform of the image
  - Then we will center the discrete Fourier transform , as we will bring the discrete Fourier transform in center from corners
  - Then we will apply filtering , means we will multiply the Fourier transform by a filter function
  - Then we will again shift the DFT from center to the corners
  - Last step would be take to inverse discrete Fourier transform , to bring the result back from frequency domain to spatial domain
  - And this step of post processing is optional , just like pre processing , in which we just increase the appearance of image.

- FILTERS
  - The concept of filter in frequency domain is same as the concept of a mask in convolution.
- After converting an image to frequency domain, some filters are applied in filtering process to perform different kind of processing on an image. The processing include blurring an image , sharpening an image e.t.c. The common type of filters for these purposes are:
  - Ideal high pass filter
  - Ideal low pass filter
  - Gaussian high pass filter
  - Gaussian low pass filter

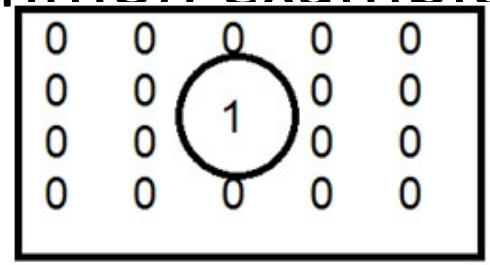
# Blurring masks vs derivative masks.

- A blurring mask has the following properties.
  - All the values in blurring masks are positive
  - The sum of all the values is equal to 1
  - The edge content is reduced by using a blurring mask
  - As the size of the mask grow, more smoothing effect will take place

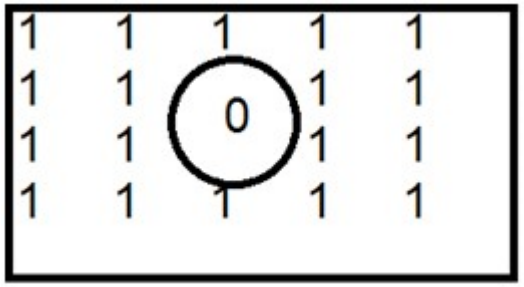
- A derivative mask has the following properties.
  - A derivative mask have positive and as well as negative values
  - The sum of all the values in a derivative mask is equal to zero
  - The edge content is increased by a derivative mask
  - As the size of the mask grows , more edge content is increased

- The relationship between blurring mask and derivative mask with a high pass filter and low pass filter can be defined simply as.
  - Blurring masks are also called as low pass filter
  - Derivative masks are also called as high pass filter

- This is the common example of low pass filter.

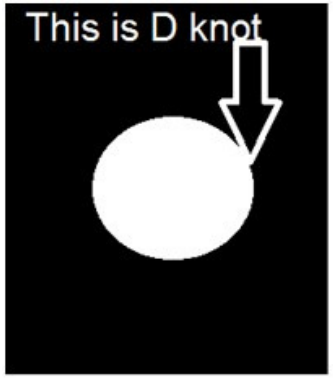



  - When one is placed inside and the zero is placed outside , we got a blurred image. Now as we increase the size of 1, blurring would be increased and the edge content would be reduced.
- This is a common example of high pass filter.

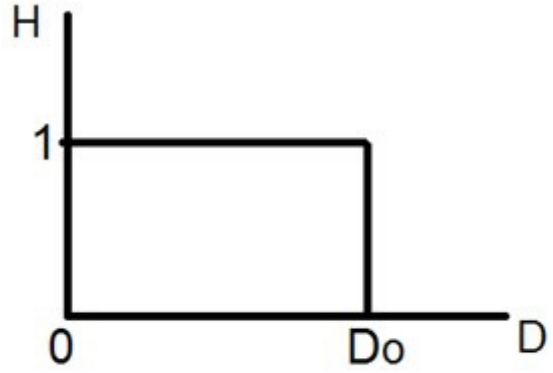



  - When 0 is placed inside, we get edges , which gives us a sketched image.

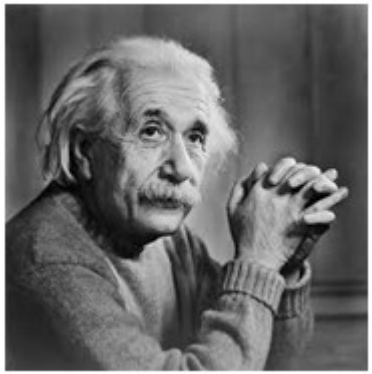
- an ideal low pass filter in frequency domain is given below

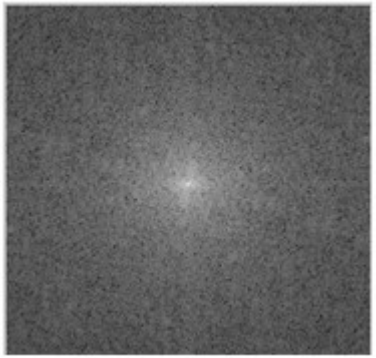



- Graphically represented by:

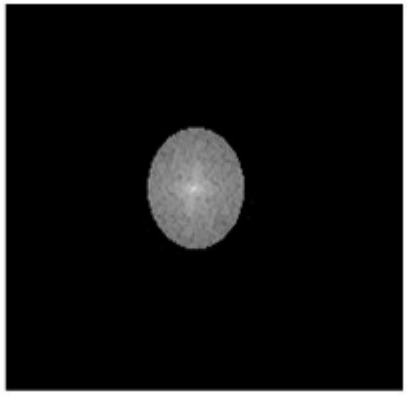


# Apply low pass

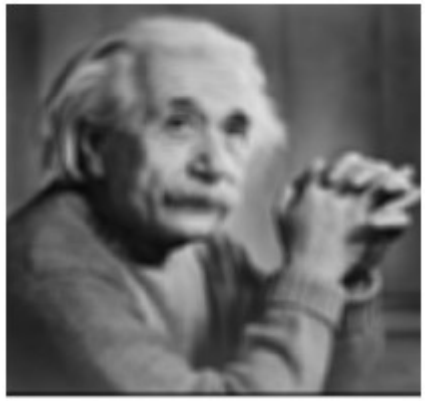




Applying filter

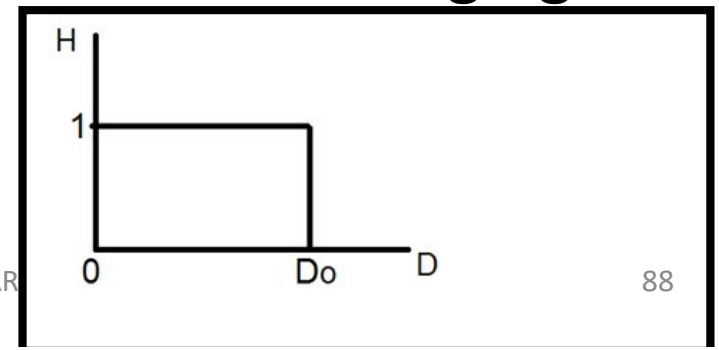


Resultant image

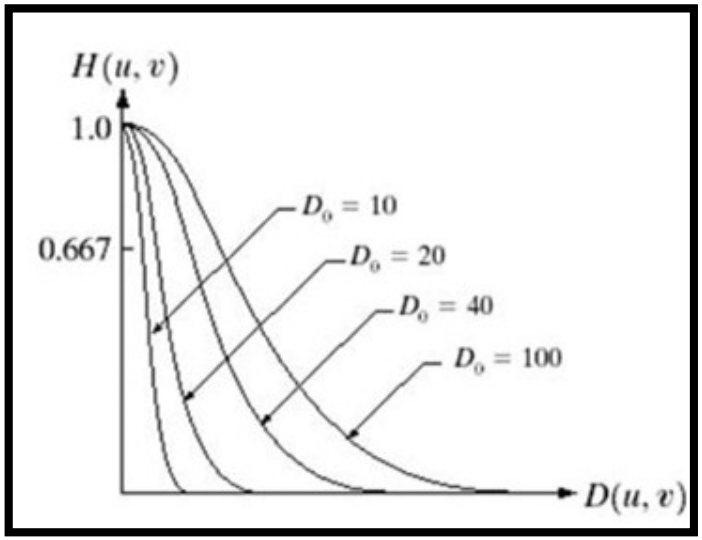


# Gaussian Low pass and Gaussian High pass filter

- Gaussian low pass and Gaussian high pass filter minimize the problem that occur in ideal low pass and high pass filter. This problem is known as ringing effect.
  - This is due to reason because at some points transition between one color to the other cannot be defined precisely, due to which the ringing effect appears at that point.
  - at the exact point of Do , you cannot tell that the value would be 0 or 1. Due to which the ringing effect appears at that point.



- GAUSSIAN LOW PASS FILTER
  - The concept of filtering and low pass remains the same, but only the transition becomes different and become more smooth. The Gaussian low pass filter can be represented as

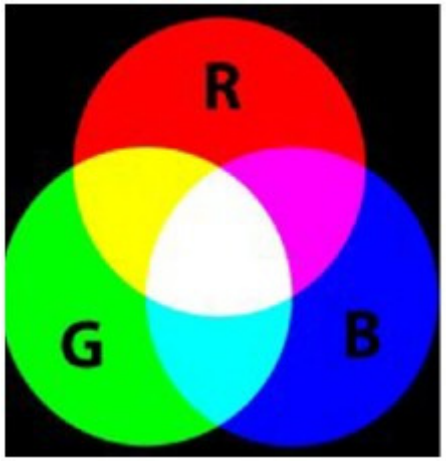



- Note the smooth curve transition, due to which at each point, the value of Do , can be exactly defined.
- GAUSSIAN HIGH PASS FILTER
  - Gaussian high pass filter has the same concept as ideal high pass filter , but again the transition is more smooth as compared to the ideal one.

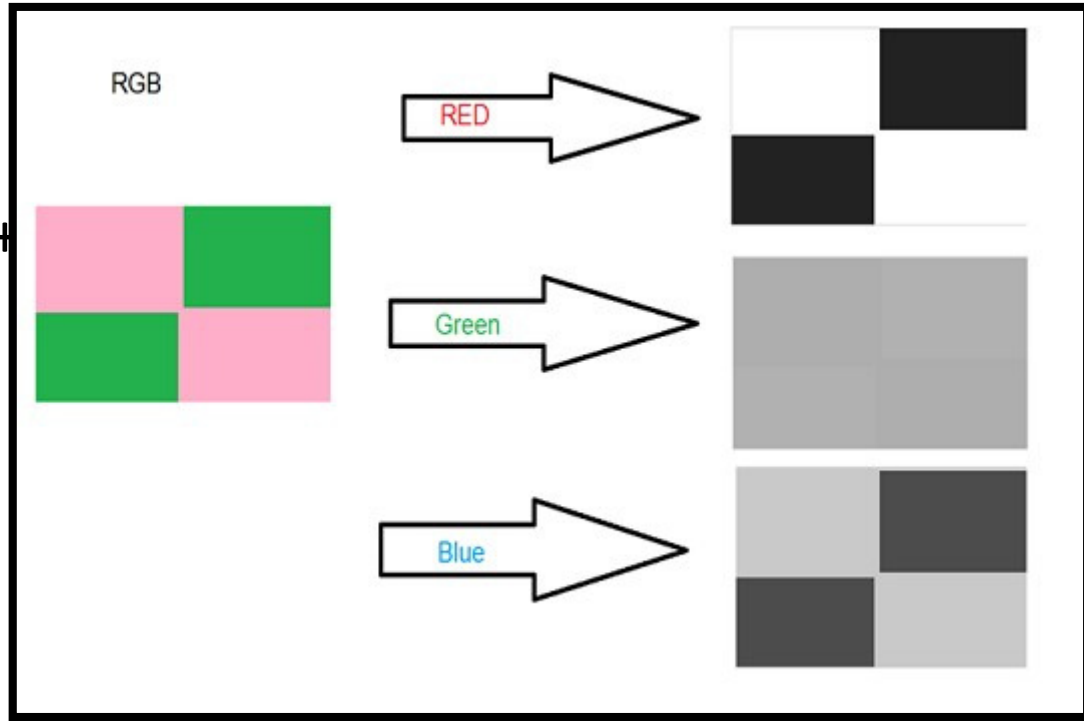
# Color Space

- Color spaces are different types of color modes, used in image processing and signals and system for various purposes. Some of the common color spaces are:
  - RGB
  - CMY'K
  - Y'UV
  - YIQ
  - Y'CbCr
  - HSV

# RGB




- A normal grayscale image can be defined by only one matrix, but a color image is actually composed of three different matrices.
- One color image matrix = red matrix + blue matrix + green matrix

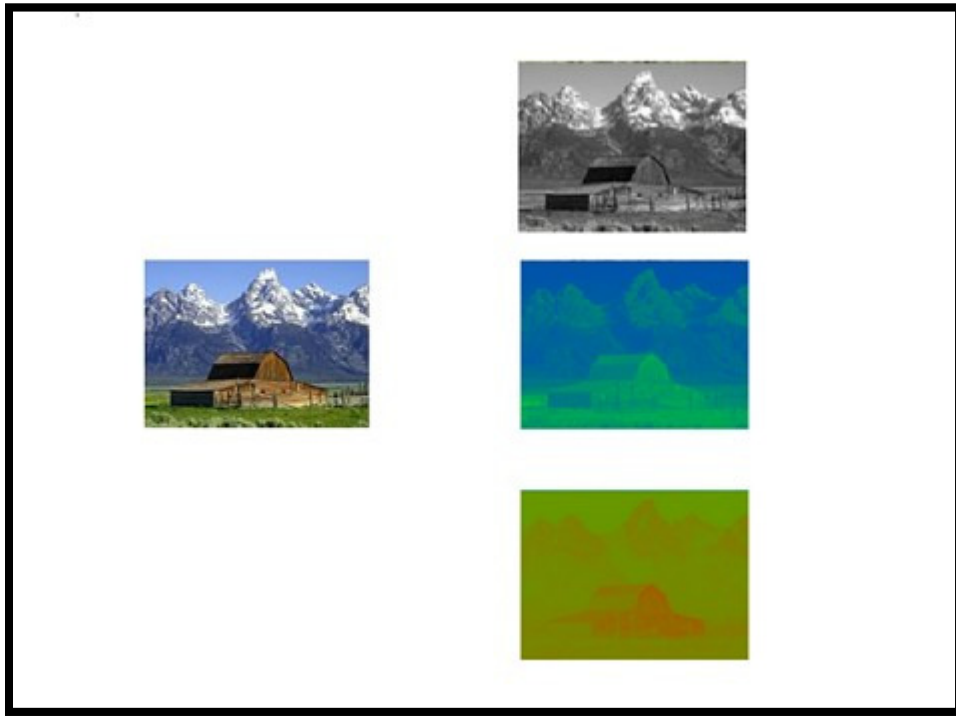


# CMYK

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 255 \\ 255 \\ 255 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

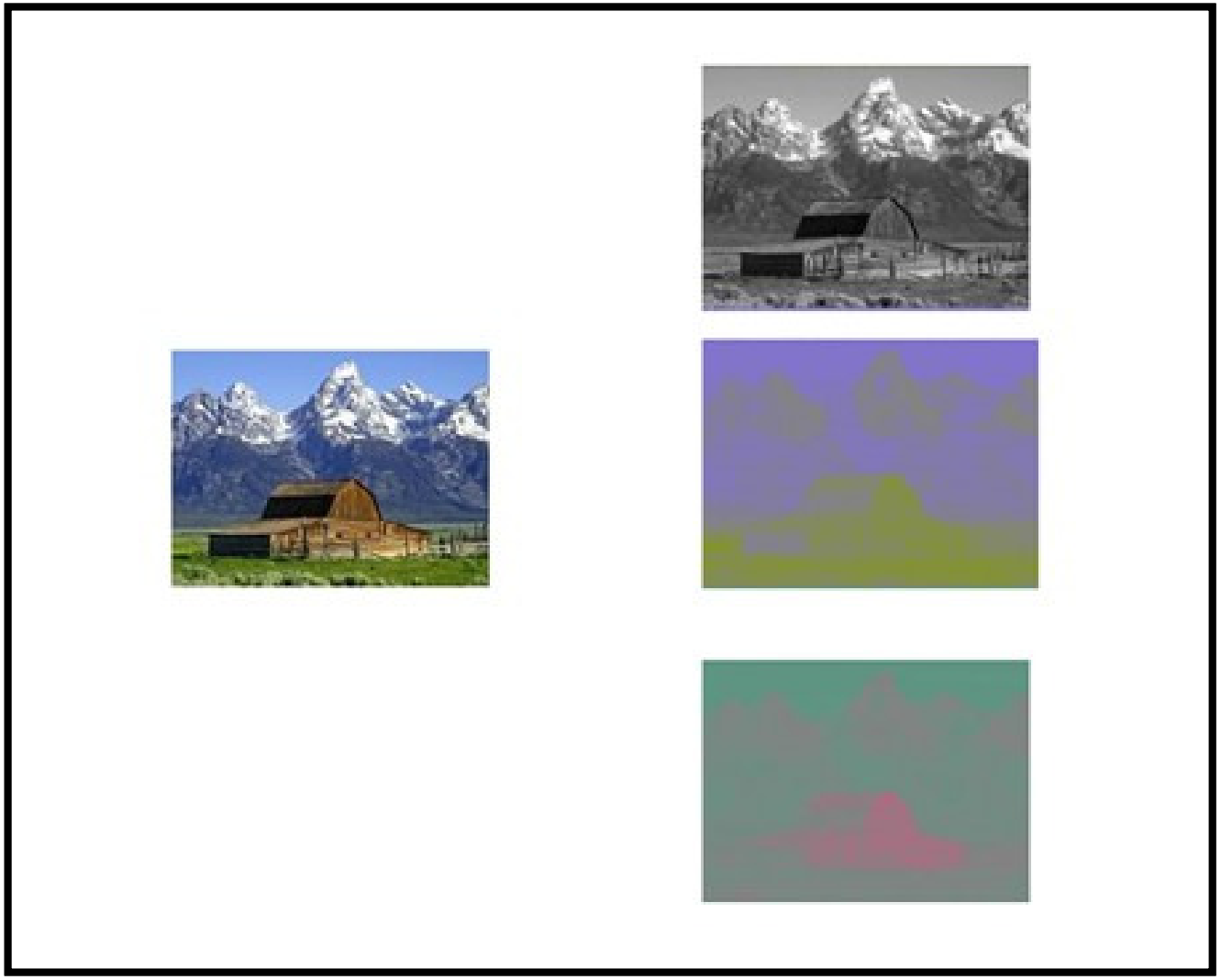
# YUV

- defines a color space in terms of one luma (Y') and two chrominance (UV) components. The Y'UV color model is used in the following composite color video standards.
  - NTSC ( National Television System Committee)
  - PAL (Phase Alternating Line)
  - SECAM (Sequential couleur a amemoire, French for "sequential color with memory)



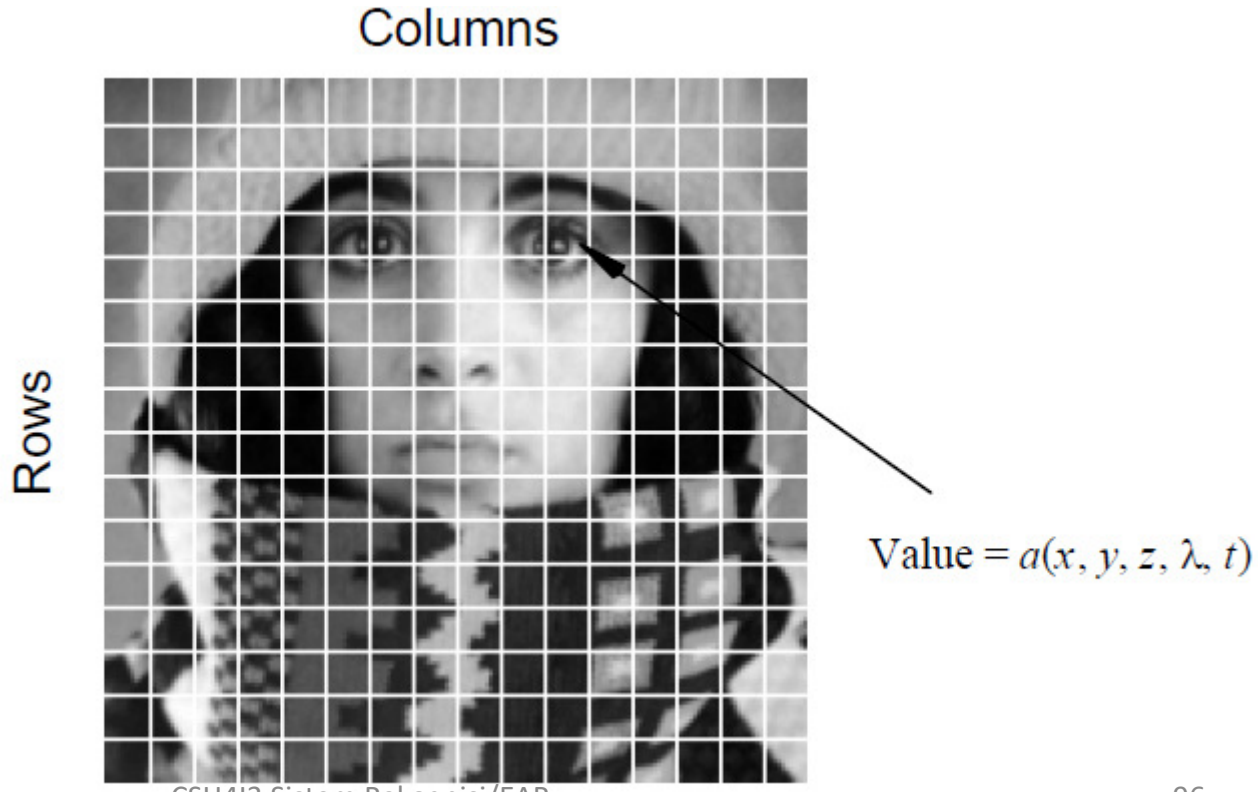
# YCBCR

- contains Y’ , the luma component and cb and cr are the blue-difference and red difference chroma components.
- It is not an absolute color space. It is mainly used for digital systems. Its common applications include JPEG and MPEG compression.
- Y’UV is often used as the term for Y’CbCr, however they are totally different formats. The main difference between these two is that the former is analog while the later is digital.

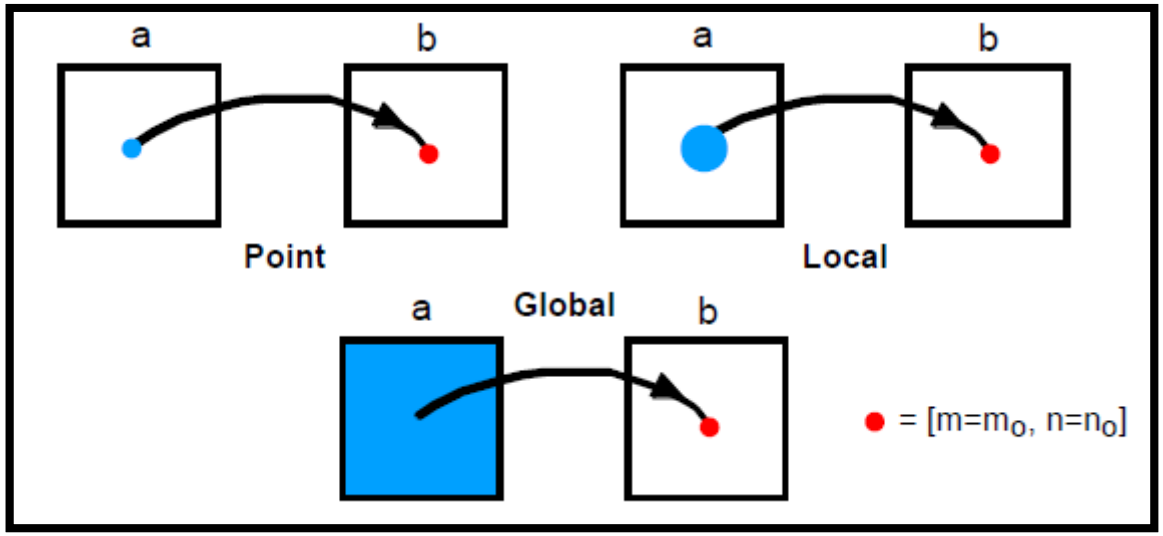


# Image Manipulation

- Image Processing — *image in → image out*
- Image Analysis — *image in → measurements out*
- Image Understanding — *image in → high-level description out*



# Image Operations




| Operation | Characterization | Generic Complexity/Pixel |
|---|---|---|
| • *Point* | – the output value at a specific coordinate is dependent only on the input value at that same coordinate. | *constant* |
| • *Local* | – the output value at a specific coordinate is dependent on the input values in the *neighborhood* of that same coordinate. | $P^2$ |
| • *Global* | – the output value at a specific coordinate is dependent on all the values in the input image. | $N^2$ |

# Type of neighborhood

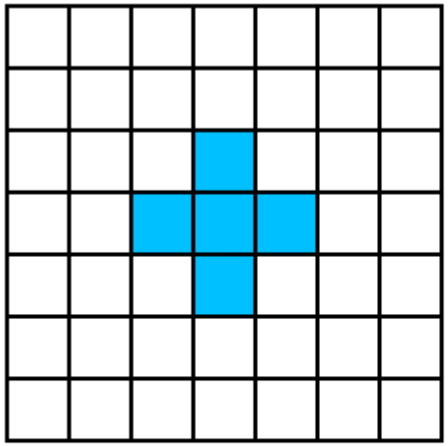


**Figure 3a**
Rectangular sampling
4-connected

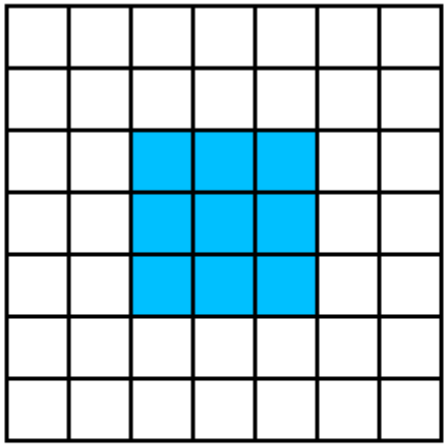


**Figure 3b**
Rectangular sampling
8-connected

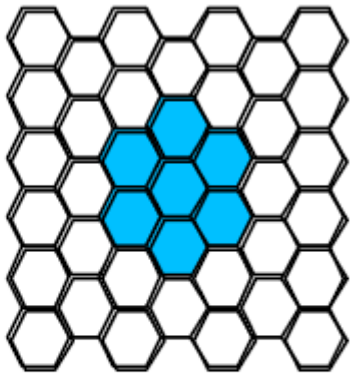
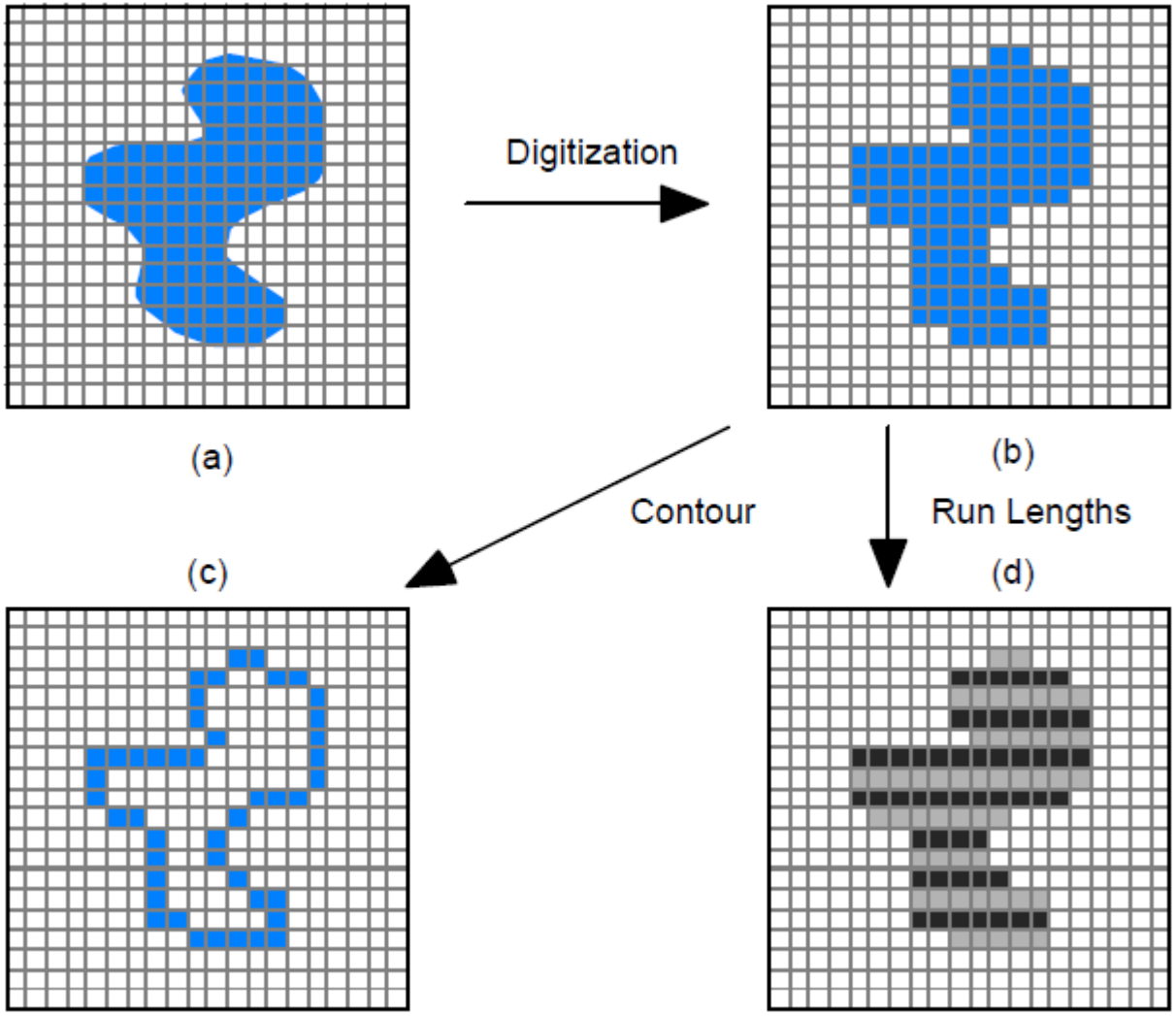


**Figure 3c**
Hexagonal sampling
6-connected

# Contour Representation



# What is a Histogram?

- In Statistics, **Histogram** is a graphical representation showing a visual impression of the distribution of data.
- An **Image Histogram** is a type of histogram that acts as a graphical representation of the lightness/color distribution in a digital image. It plots the number of pixels for each value.

# Histogram Processing

- The histogram of a digital image with gray levels in the range [0, L-1] is a discrete function $h(r_k) = n_k$, where $r_k$ is the kth gray level and $n_k$ is the number of pixels in the image having gray level $r_k$.

# Histogram Processing

- It is common practice to normalize a histogram by dividing each of its values by the total number of pixels in the image, denoted by n. Thus, a normalized histogram is given by $p(r_k) = n_k / n$, for k = 0, 1, …, L -1.
- Thus, $p(r_k)$ gives an estimate of the probability of occurrence of gray level $r_k$. Note that the sum of all components of a normalized histogram is equal to 1.

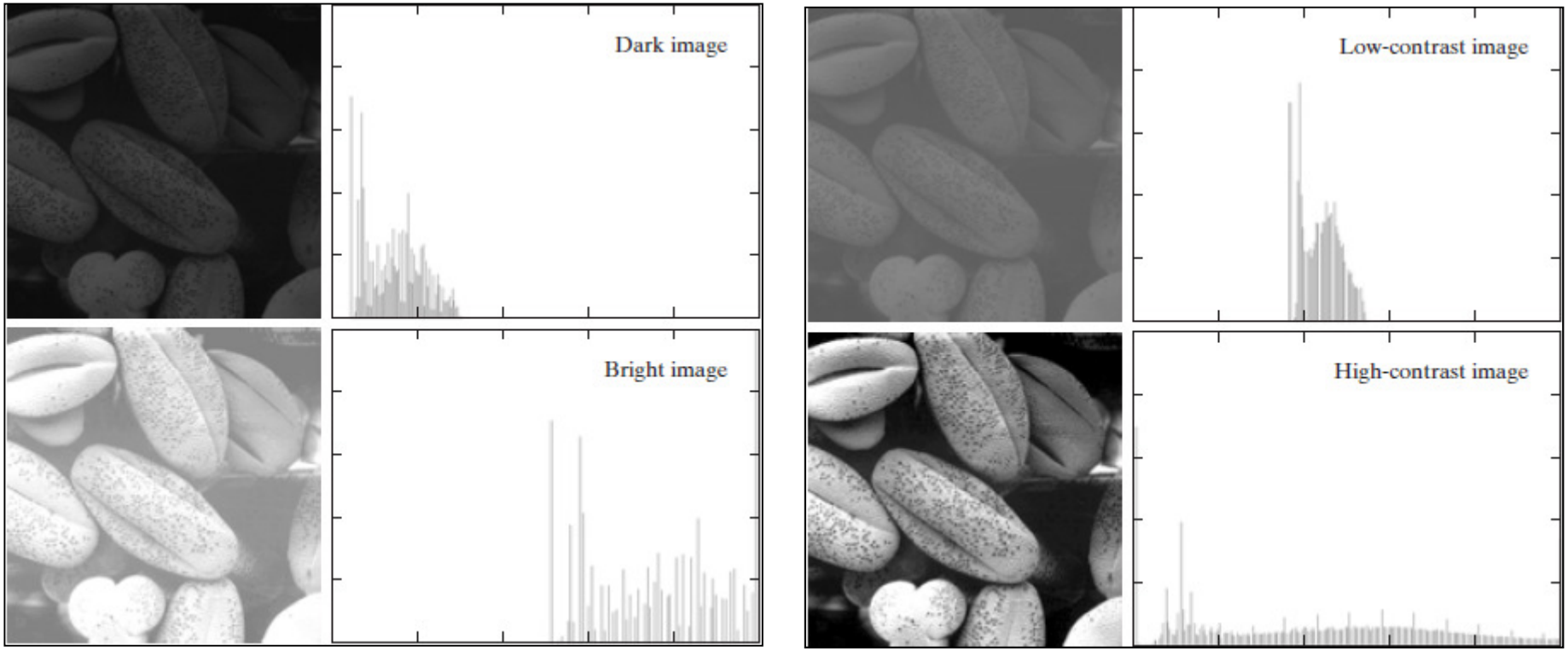
# Why Histogram?

- Histograms are the basis for numerous spatial domain processing techniques
- Histogram manipulation can be used effectively for image enhancement
- Histograms can be used to provide useful image statistics
- Information derived from histograms are quite useful in other image processing applications, such as image compression and segmentation.

# Introductory Example of Histograms

- As an introduction to the role of histogram processing in image enhancement, consider Fig. 3.15 shown in four basic gray-level characteristics: dark, light, low contrast, and high contrast.
- The right side of the figure shows the histograms corresponding to these images.
- The horizontal axis of each histogram plot corresponds to gray level values, $r_k$.
- The vertical axis corresponds to values of $h(r_k)=n_k$ or $p(r_k)=n_k/n$ if the values are normalized.
- Thus, as indicated previously, these histogram plots are simply plots of $h(r_k)=n_k$ versus $r_k$ or $p(r_k)=n_k/n$ versus $r_k$.

# Introductory Example of Histograms… Cont.



# Introductory Example of Histograms… Cont.

We note in the dark Image that the components of the histogram are concentrated on the low (dark) side of the gray scale. Similarly, the components of the histogram of the bright image are biased toward the high side of the gray scale. An image with low contrast has a histogram that will be narrow and will be centered toward the middle of the gray scale. For a monochrome image this implies a dull, washed-out gray look. Finally, we see that the components of the histogram in the high-contrast image cover a broad range of the gray scale and, further, that the distribution of pixels is not too far from uniform, with very few vertical lines being much higher than the others. Intuitively, it is reasonable to conclude that an image whose pixels tend to occupy the entire range of possible gray levels and, in addition, tend to be distributed uniformly, will have an appearance of high contrast and will exhibit a large variety of gray tones.

# Histogram in MATLAB

**h = imhist (f, b)**

Where f, is the input image, h is the histogram, b is number of bins (tick marks) used in forming the histogram (b = 255 is the default)

A bin, is simply, a subdivision of the intensity scale. For example, if we are working with uint8 images and we let b = 2, then the intensity scale is subdivided into two ranges: 0 – 127 and 128 – 255. the resulting histograms will have two values: h(1) equals to the number of pixels in the image with values in the interval [0,127], and h(2) equal to the number of pixels with values in the interval [128  255].

# Histogram in MATLAB

- We obtain the normalized histogram simply by using the expression.

  **p = imhist (f, b) / numel(f)**

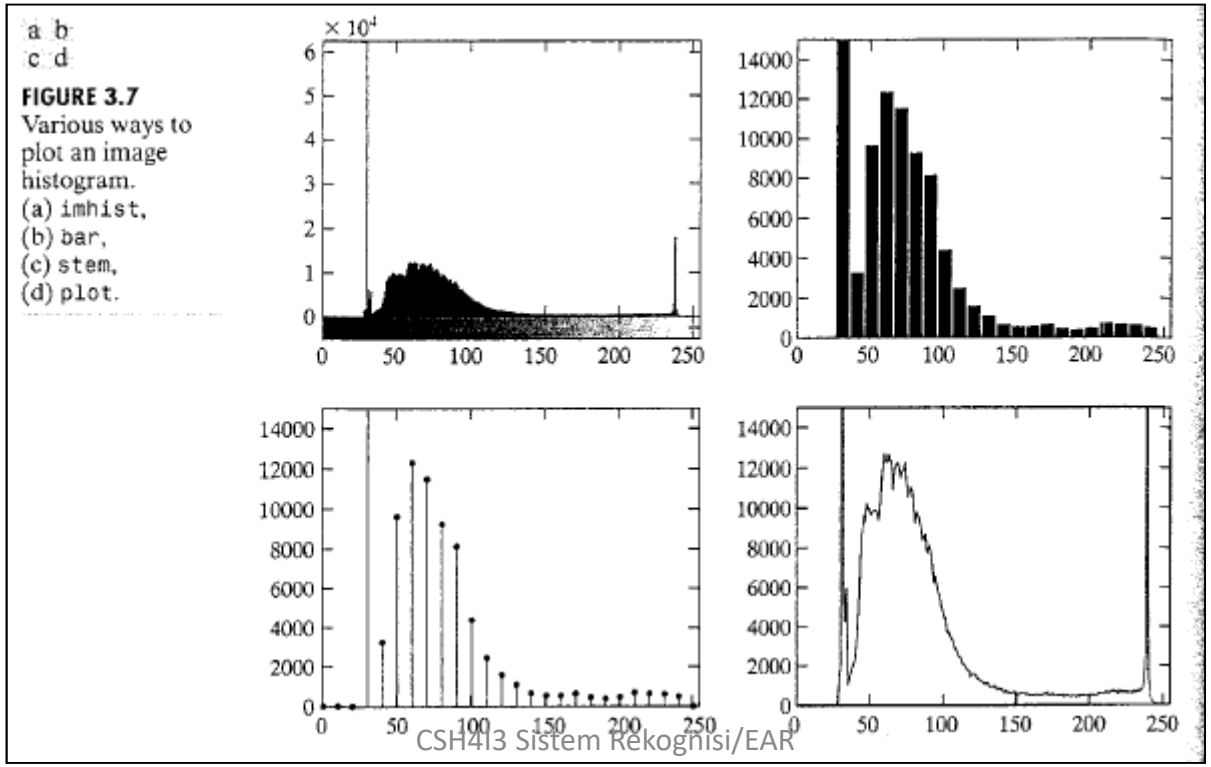
  **numel (f)**: a MATLAB function that gives the number of elements in array f (i.e. the number of pixels in an image.

# Other ways to display Histograms

- Consider an image f. The simplest way to plot its histogram is to use imhist with no output specified:

  >> imhist (f);

  Figure 3.7(a) shows the result.



# Other ways to display Histograms

- The histogram displayed in figure 3.7(a), is the default histogram in MATLAB
- However, there are other ways to plot a histogram.
- Histograms often are plotted using bar graphs. For this purpose we can use function

**bar (horz, v, width)** **(1)**

where v is a row vector containing the points to be plotted, horz is a vector of the same dimension as v that contains the increments of the horizontal scale, and width is a number between 0 and 1.

if horz is omitted, the horizontal axis is divided in units from 0 to length(v). When width is 1 the bars touch; when it is 0 the bars are simply vertical lines. The default value is 0.8.

# Other ways to display Histograms

- The following statements produce a bar graph, with the horizontal axis divided into groups of 10 levels:

  **>> h = imhist(f);**

  **>>h1 = h(1:10:256);**

  **>> horz = 1:10:256;**

  **>> bar (horz, h1);**

  **>> axis ([0 255 0 15000])**

  **>> set (gca, 'xtick', 0:50:255)**

  **>>set (gca, 'ytick', 0:2000:15000)**

Figure 3.7(b) shows the result.

The **axis** function has the syntax:

**axis ([horizmin horzmax vertmin vertmax])**

Which sets the minimum and maximum values in the horizontal and vertical axes.

# Other ways to display Histograms

- **gca:** get current axes (the axes of the figure last displayed)
- **xtick** and **ytick** set the horizontal and vertical ticks in the interval shown.
- Axes labels can be added to the horizontal and vertical axes of a graph using the functions

  **xlabel** (‘text string’, ‘fontsize’, size);

  **ylabel** (‘text string’, ‘fontsize’, size);

  Where size is the font size in points

- Text can be added to the body of the figure using function:

  **text** (xloc, yloc, ‘text string’, ‘font size’, size);

  where: xloc and yloc define the location where text starts.

# Other ways to display Histograms

- **Note: functions that set axis values and labels are used after the function has been plotted.**

- A title can be added to a plot using:

  **title**(‘text string’)

# Other ways to display Histograms

- A **stem** graph can be used to display the histogram:
  **stem (horz, v, ‘fill’, ‘LineStyle’);** **(2)**

  where v and horz as in **bar** function.

  where ‘LineStyle’ represents the shape of lines in the graph

  if ‘fill’ is used, and the marker is circle, square or diamond, the marker is filled, with the color specified. The default color is black.
- To set the shape and the color of the marker, use the following Set statements (example):
- >> stem (horz, v, ‘fill’, ‘- -’);
- stem(horz, v, 'fill', '', 'color', 'r', 'marker', 's');
- >> set (h, ‘MarkerFaceColor’, ‘r’, ‘marker’, ‘s’); ///OLD MATLAB Version
- This sentence will change the shape of the marker to red square.

# Other ways to display Histograms

| Symbol | Color | Symbol | Line Style | Symbol | Marker |
|---|---|---|---|---|---|
| k | Black | - | Solid | + | Plus sign |
| w | White | -- | Dashed | o | Circle |
| r | Red | : | Dotted | * | Asterisk |
| g | Green | -. | Dash-dot | . | Point |
| b | Blue | none | No line | x | Cross |
| c | Cyan | | | s | Square |
| y | Yellow | | | d | Diamond |
| m | Magenta | | | none | No marker |

**TABLE 3.1**
Attributes for functions `stem` and `plot`. The `none` attribute is applicable only to function `plot`, and must be specified individually. See the syntax for function `plot` below.

# Other ways to display Histograms

- **Plot graph**, can be used to display the histogram with straight lines, the syntax is:

**h = imhist(f);**

**plot (h, 'color-linestyle-marker')** **(3)**

the arguments are specified preciously.

# References