University of Baghdad Fourth Class
College of education Ass. Prof. Dr. Hussein Al-Kaabi

Digital Image processing

Chapter One

Introduction to Computer Vision and Image Processing

1.1. __What Is Digital Image Processing?__

An image may be defined as a two-dimensional function, $f(x, y)$,where x and y are *spatial* (plane) coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the *intensity* or *gray level* of the image at that point.

When x, y, and the intensity values of f are all finite, discrete quantities, we call the image a *digital image*.

Note that a digital image is a representation of a two-dimensional image as a finite set of digital values composed of a finite number of elements, each of which has a particular location and value.These elements are called *picture elements*, *image elements*, and *pixels*. *Pixel* is the term used most widely to denote the elements of a digital image

__Note: One picture is worth more than ten thousand words.__

1.2 The Human Visual System

The Human Visual System (HVS) has two primary components:

- Eye.
- Brian.

The structure that we know the most about is the image receiving sensors (the human eye).

The brain can be thought as being an information-processing unit analogous to the computer in our computer imaging system. These two are connected by the optic nerve, which is really a bundle of nerves that contains the pathways for visual information to travel from the receiving sensor (the eye) to the processor (the brain).

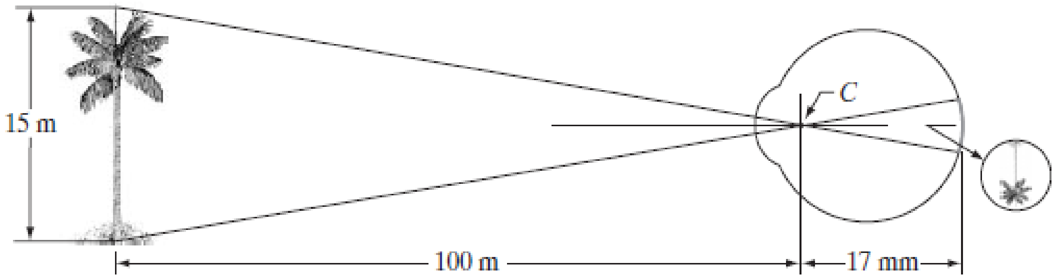


FIGURE 1.1 An example of the human image acquisition process.

1.3. Digital image acquisition process

Digitization The process of transforming a standard video signal into digital image. This transformation is necessary because the standard video signal in analog (continuous) form and the computer requires a digitized or sampled version of that continuous signal. The analog video signal is turned into a digital image by sampling the continuous signal at affixed rate. In the figure below we see one line of a video signal being sampled (digitized) by instantaneously measuring the voltage of the signal (amplitude) at fixed intervals in time.

The value of the voltage at each instant is converted into a number that is stored, corresponding to the brightness of the image at that point. Note that **the image brightness of the image at that point** depends on both the intrinsic properties of the object and the lighting conditions in the scene.

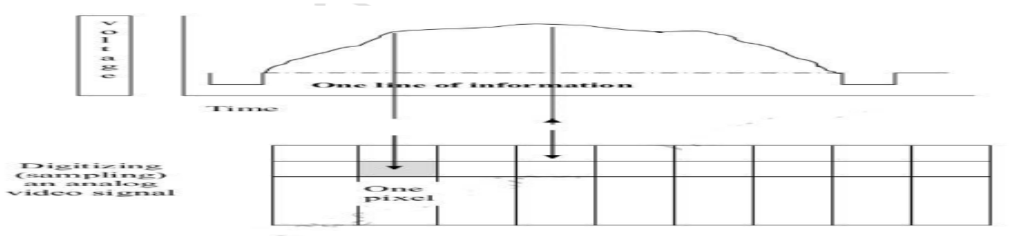


Figure (1.2) Digitizing (Sampling) an Analog Video Signal

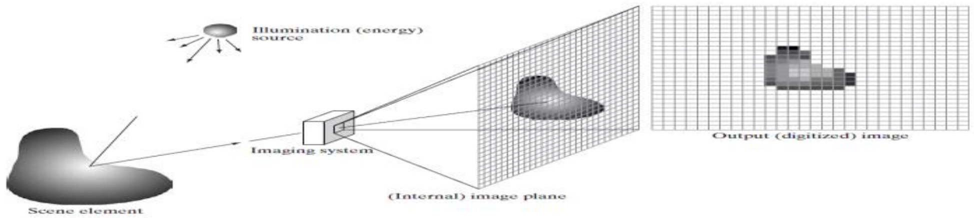


FIGURE 1.3 An example of the digital image acquisition process. (a) Energy (“illumination”) source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

The image can now be accessed as a two-dimension array of data , where each data point is referred to a pixel (picture element).for digital images we will use the following notation :

I(r,c) = The brightness of image at the point (r,c) Where r= row and c= column.

“When we have the data in digital form, we can use the software to process the data”. The digital image is 2D- array as:

$$\begin{pmatrix} I(0,0) & I(0,1) & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & I(0,N-1) \\ I(1,0) & I(1,1) & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & I(1,N-1) \\ \ldots\ldots & \ldots\ldots & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & \ldots\ldots \\ \ldots\ldots & \ldots\ldots & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & \ldots\ldots \\ I(N-1,0) & I(N-1,1) & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & I(N-1,N-1) \end{pmatrix}$$

In above image matrix, the image size is (N×N) [matrix dimension] then:

$Ng= 2^{m}$. Where Ng denotes the number of gray levels m, where m is the no. of bits contains in digital image matrix.

Example : If we have (6 bit) in 128 * 128 image .Find the |No. of gray levels to represent it ,then find the no. of bit in this image?

Solution:

$N_g= 2^{6}=64$ Gray Level

$N_b= 128 * 128* 6 = 9.8304 * 10^{4}$ bit

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too.

1.4. Image Representation

We have seen that the **H**uman **V**isual **S**ystem (HVS) receives an input image as a collection of spatially distributed light energy; this is form is called an optical image. Optical images are the type we deal with every day –cameras captures them, monitors display them, and we see them [we know that these optical images are represented as video information in the form of analog electrical signals and have seen how these are sampled to generate the digital image I(r , c).

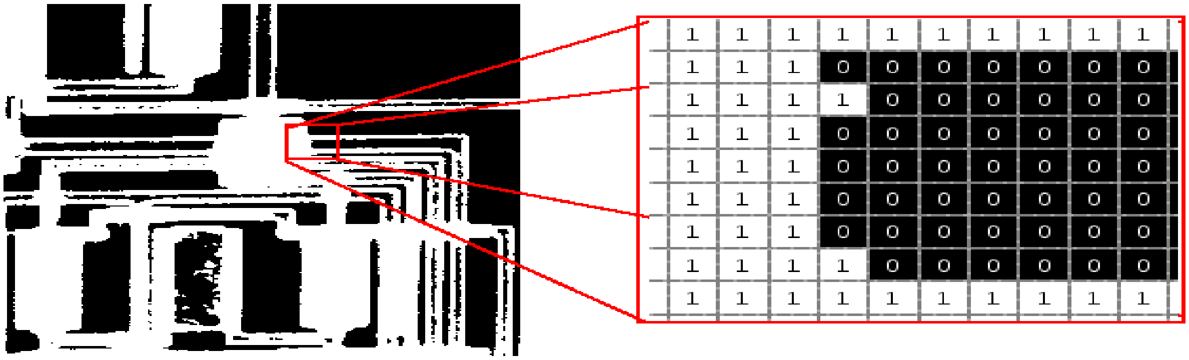
The digital image I (r, c) is represented as a two- dimensional array of data, where each pixel value corresponds to the brightness of the image at the point (r, c). in linear algebra terms , a two-dimensional array like our image model I(r, c) is referred to as a matrix , and one row (or column) is called a vector.

The image types we will consider are:

1.4.1. Binary Image

Binary images are the simplest type of images and can take on two values, typically black and white, or ‘0’ and ‘1’. A binary image is referred to as a 1 bit/pixel image because it takes only 1 binary digit to represent each pixel.

These types of images are most frequently in computer vision application where the only information required for the task is general shapes, or outlines information. For example, to position a robotics gripper to grasp an object or in optical character recognition (OCR). Binary images are often created from gray-scale images via a threshold value is, those values above it are turned white (‘1’), and those below it are turned black (‘0’).



1	1	1	1	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1

Figure (1.4): Binary Images.

1.4.2. Gray-scale images.

Gray-scale image is a range of monochromatic shades from black to white. Therefore, a grayscale image contains only shades of gray (brightness information only) and no color information. The number of different brightness level available. (0) value refers to black color, (255) value refers to white color, and all intermediate values are different shades of gray varying from black to white. The typical image contains 8 bit/ pixel (data, which allows us to have (0-255) different brightness (gray) levels. The 8 bit representation is typically due to the fact that the byte, which corresponds to 8-bit of data, is the standard small unit in the world of digital computer.

Figure (1.5) : Gray Scale Image

1.4.3. Color image

Color image can be modeled as three band monochrome image data, where each band of the data corresponds to a different color. The actual information stored in the digital image data is brightness information in each spectral band. When the image is displayed, the corresponding brightness information is displayed on the screen by picture elements that emit light energy corresponding to that particular color. Typical color images are represented as red, green, and blue or RGB images. Using the 8-bit monochrome standard as a model, the corresponding color image would have 24 bit/pixel – 8 bit for each color bands (red, green and blue). The following figure we see a representation of a typical RGB color image.

IR(r,c) IG(r,c) IB(r,c).

The following figure illustrate that in addition to referring to arrow or column as a vector, we can refer to a single pixel red ,green, and blue values as a color pixel vector –(R,G,B).

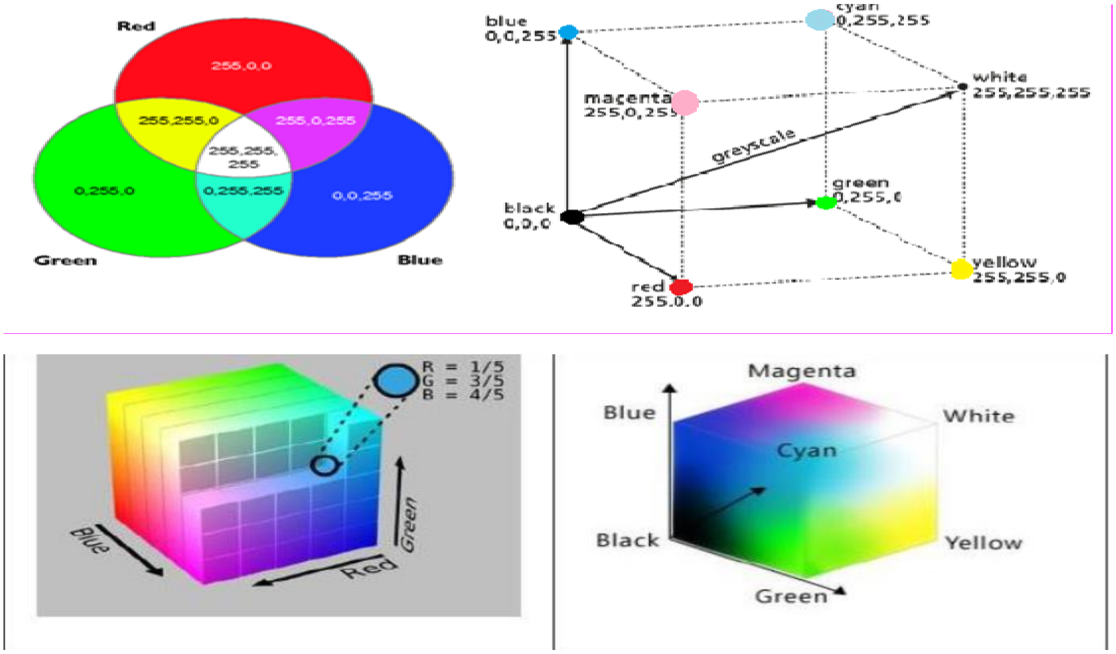


Figure (1.6): a color pixel vector consists of the red, green and blue pixel values (R, G, B) at one given row/column pixel coordinate (r,c).

The RGB color model used in color CRT monitor, in this model ,Red,Grren and Blue are added together to get the resultant color White.

Red	Green	Blue	Color
0	0	0	Black
0	0	1	Blue
0	1	0	Green
1	0	0	Red
0	1	1	Cyan
1	0	1	Magenta
1	1	0	Yellow
1	1	1	White

For many applications, RGB color information is transformed into mathematical space that decouples the brightness information from the color information.

The lightness is the brightness of the color, and the hue is what we normally think of as “color” and the hue (ex: green, blue, red, and orange). The saturation is a measure of how much white is in the color (ex: Pink is red with more white, so it is less saturated than a pure red). Most people relate to this method for describing color.

Example: “a deep, bright orange” would have a large intensity (“bright”), a hue of “orange”, and a high value of saturation (“deep”).we can picture this color in our minds, but if we defined this color in terms of its RGB components, R=245, G=110 and B=20, most people have no idea how this color appears. Modeling the color information creates a more people oriented way of describing the colors.

1.4.4 . Multispectral images

Multispectral images typically contain information outside the normal human perceptual range. This may include infrared (تحت الحمراء),ultraviolet (فوق البنفسجية), X-ray, acoustic or radar data. These are not images because the information represented is not directly visible by the human system. Source of these types of image include satellite systems, underwater sonar systems and medical diagnostics imaging systems.

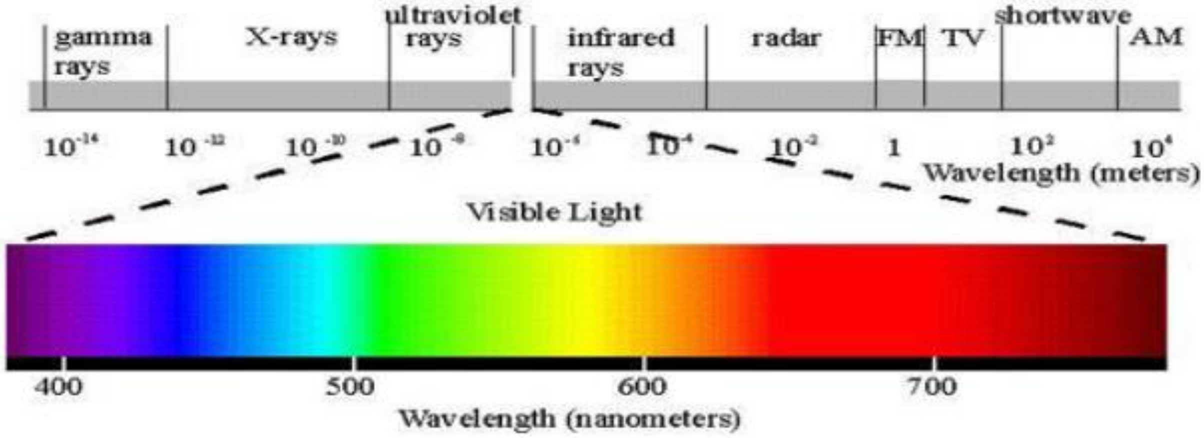


Figure (1.7) Electromagnetic spectrum.

1.5. Image Processing Applications

Image processing systems are used in many and various types of environments, such as:

1. Medical community has many important applications for image processing involving various type diagnostics imaging , as an example Magnetics Resonance Imaging scanning(MRI), that allows the medical professional to look into the human body without the need to cut it open, CT scanning, X-RAY imaging .

2. Computer – Aided Design(CAD), which uses tools from image processing and computer graphics, allows the user to design a new building or spacecraft and explore it from the inside out.

3. Virtual Reality is one application that exemplifies (يمثل) future possibilities

4. Machine/Robot vision: Make robot able to see things , identify them , identify the hurdles.

1.6 Computer Imaging Systems

Computer imaging systems are comprised of two primary components types, hardware and software. The hard ware components can be divided into image acquiring sub system (computer, scanner, and camera) and display devices (monitor, printer).The software allows us to manipulate the image and perform any desired processing on the image data.

1.7. Image Resolution

Pixels are the building blocks of every digital image. Clearly defined squares of light and color data are stacked up (مكدسة) next to one another both horizontally and vertically. Each **picture element (pixel for short)** has a dark to light value from 0 (solid black) to 255 (pure white).

That is, there are 256 defined values. **A gradient** (ميل، نسبة الانحدار) is the gradual transition from one value to another in sequence.

The resolution has to do with ability to separate two adjacent pixels as being separate, and then we can say that we can resolve the two. The concept of resolution is closely tied to the concepts of spatial frequency. There are three types of image resolution:

1- ***Vertical resolution***: the number of M rows in the image (image scan line).

2- ***Horizontal resolution***: the number of N columns in the image.

3- ***Spatial frequency resolution***: It is represent by the multiplication (M x N) and its closely tied to the concept of spatial frequency that refers to how rapidly the signal is changing in space, and the signal has two values for brightness 0 and maximum. If we use this signal for one line (row) of an image and then repeat the line down the entire image, we get an image of vertical stripes. If we increase this frequency the strips get closer and closer together, until they finally blend together as shown in figure below, note that the higher the resolution the more details (high frequency).

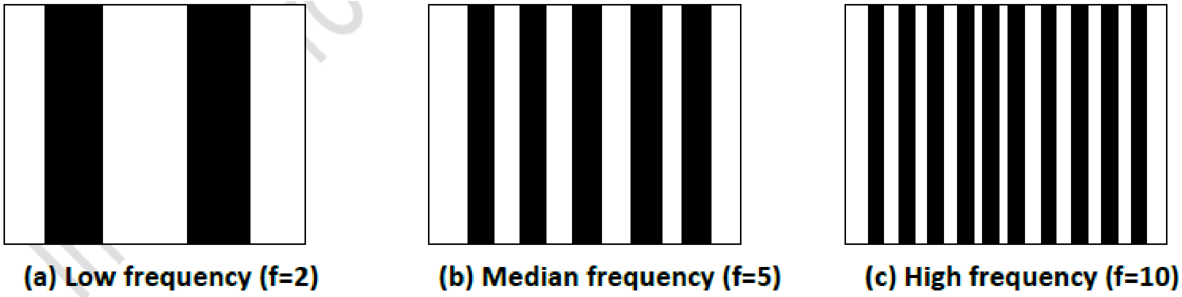

(a) Low frequency (f=2)
(b) Median frequency (f=5)
(c) High frequency (f=10)

Figure (1.8) Resolution and Spatial frequency.

In computers, resolution is the number of pixels (individual points of color) contained on a display monitor, expressed in terms of the number of pixels on the horizontal axis and the number on the vertical axis.

The sharpness of the image on a display depends on the resolution and the size of the monitor. The same pixel resolution will be sharper on a smaller monitor and gradually lose sharpness on larger monitors because the same numbers of pixels are being spread out over a larger number of **inches**. Display resolution is not measured in dots per inch as it usually is with printers ***(*****We measure resolution in pixels per inch or more commonly, dots per inch (dpi)).**

In computers, resolution is the number of pixels (individual points of color) contained on a display monitor, expressed in terms of the number of pixels on the horizontal axis and the number on the vertical axis.

The sharpness of the image on a display depends on the resolution and the size of the monitor.

Display resolution is not measured in dots per inch as it usually is with printers *(We measure resolution in pixels per inch or more commonly, dots per inch (dpi)).*

- A display with 240 pixel columns and 320 pixel rows would generally be said to have a resolution of **240×320.**
- Resolution can also be used to refer to the total number of pixels in a digital camera image. For example, a camera that can create images of 1600x1200 pixels will sometimes be referred to as a 2 megapixel resolution camera since 1600 x 1200 = 1,920,000 pixels, or roughly 2 million pixels. Where a megapixel (that is, a million pixels) is a unit of image sensing capacity in a digital camera. In general, the more megapixels in a camera, the better the resolution when printing an image in a given size.

Below is an illustration of how the same image might appear at different pixel resolutions, if the pixels were poorly rendered as sharp squares (normally, a smooth image reconstruction from pixels would be preferred, but for illustration of pixels, the sharp squares make the point better).

An image that is 2048 pixels in width and 1536 pixels in height has a total of 2048×1536 = 3,145,728 pixels or 3.1 megapixels. One could refer to it as 2048 by 1536 or a 3.1-megapixel image.

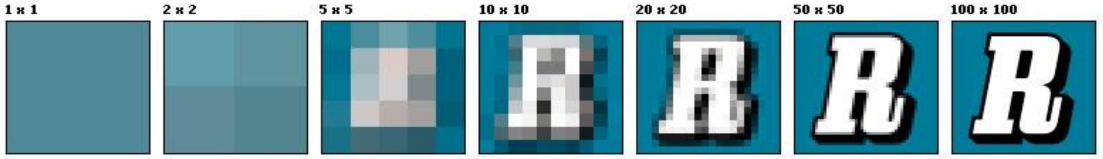


Figure (1. 9) : Image Resolution.

1.8. Computer Imaging

Can be defined a acquisition and processing of visual information by computer. Computer representation of an image requires the equivalent of many thousands of words of data, so the massive amount of data required for image is a primary reason for the development of many sub areas with field of computer imaging, such as image compression and segmentation .Another important aspect of computer imaging involves the-ultimate “receiver” of visual information in some case the human v~~i~~sual system and in some cases the human visual system and in others the computer itself.

Computer imaging can be separate into two primary categories:

1. **Computer Vision.** **2. Image Processing.**

(In computer vision application the processed images output for use by a computer, whereas in image processing applications the output images are for human consumption).

These two categories are not totally separate and distinct. The boundaries that separate the two are fuzzy, but this definition allows us to explore the differences between the two and to explore the difference between the two and to understand how they fit together (Figure 1.1).

Computer imaging can be separated into two different but overlapping areas.

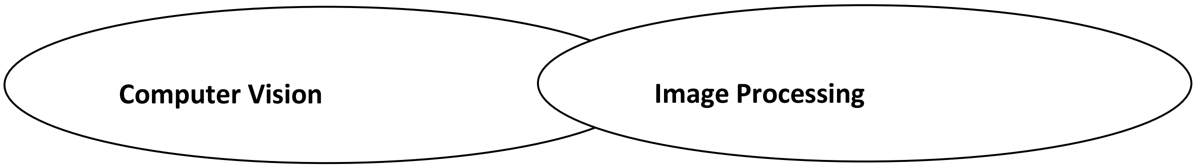


Figure (1.10): Computer Imaging .

Historically, the field of image processing grew from electrical engineering as an extension of the signal processing branch, whereas are the computer science discipline was largely responsible for developments in computer vision.

1.8.1. Computer Vision

Computer vision emulate human vision, that's mean: understanding the scene based on image data. One of the major topics within this field of computer vision is image analysis.

Image Analysis: involves the examination of the image data to facilitate solving vision problem.

The image analysis process involves two other topics:

- Feature Extraction: is the process of acquiring higher level image information, such as shape or color information.
- Pattern Classification: is the act of taking this higher –level information and identifying objects within the image.

Computer vision systems are used in many and various types of environments, such as:

1. Manufacturing Systems: computer vision is often used for quality control, where the computer vision system will scan manufactured items for defects, and provide control signals to a robotics manipulator to remove detective part automatically.

2. Medical Community: current example of medical systems to aid neurosurgeons during brain surgery, systems to diagnose skin tumors automatically.

3. The field of Law Enforcement (تنفيذ القانون) and security in an active area for computer vision system development, with application ranging from automatic identification of fingerprints to DNA analysis.

4. Infrared Imaging

5. Satellites Orbiting .

1.8.2. Image Processing

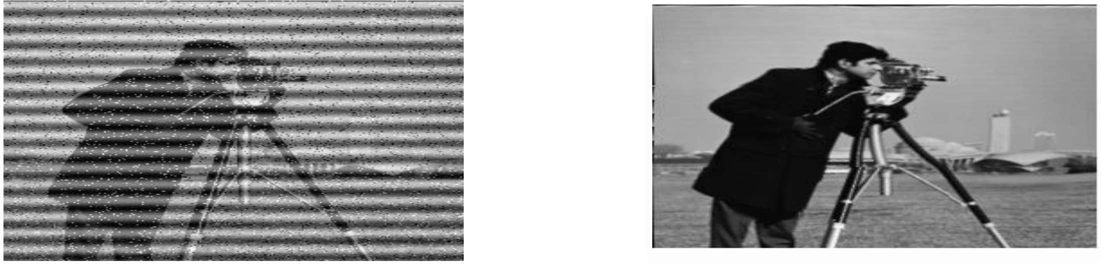
Image processing does some transformations on image. That means may be it does some smoothing, sharpening, contrasting, stretching on the image for making image more enhancive & readable that is input and output of a process are images.. In other words the image are to be examined and a acted upon by people.

The major topics within the field of image processing include:

1. Image restoration.
2. Image enhancement.
3. Image compression.

1.8.3. Image Restoration

Is the process of taking an image with some known, or estimated degradation, and restoring it to its original appearance. Image restoration is often used in the field of photography or publishing where an image was somehow degraded but needs to be improved before it can be printed (Figure 1.2).



a. Image with distortion **b. Restored image**

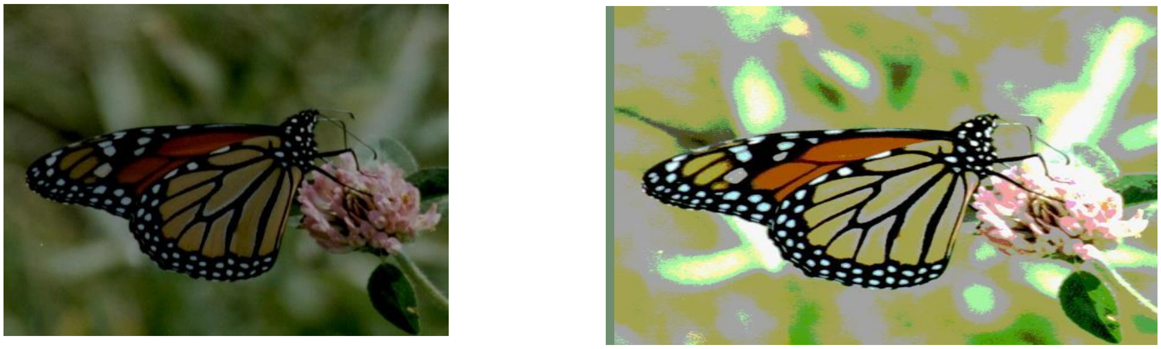
Figure (1.10) Image Restoration

1.8.4. Image Enhancement

Involves taking an image and improving it visually, typically by taking advantages of human Visual Systems responses. One of the simplest enhancement techniques is to simply stretch the contrast of an image.

Enhancement methods tend to be problem specific. For example, a method that is used to enhance satellite images may not suitable for enhancing medical images.

Although enhancement and restoration are similar in aim, to make an image look better. Restoration method attempt to model the distortion to the image and reverse the degradation, where enhancement methods use knowledge of the human visual systems responses to improve an image visually.



a. image with poor contrast **b. Image enhancement by contrast stretching**

Figure (1.11) Image Enhancement

1.8.5. Image Compression

Involves reducing the typically massive amount of data needed to represent an image. This done by eliminating data that are visually unnecessary and by taking advantage of the redundancy that is inherent

in most images. Image data can be reduced 10 to 50 times, and motion image data (video) can be reduced by factors of 100 or even 200.

a. Image before compression.

92KB

b. Image after compression.

6.59 KB

1.9. Digital Image File Format

Why do we need so many different types of image file format?

• *The short answer* is that there are many different types of images and application with varying requirements.

• *A more complete answer*, also considers market share proprietary information, and a lack of coordination within the imaging industry. Many image types can be converted to one of other type by easily available image conversion software. Field related to computer imaging is that computer graphics.

1.9.1 Computer Graphics:

Computer graphics is a specialized field within that refers to the computer science realm that refers to the reproduction of visual data using computer. In computer graphics, types of image data are divided into two primarily categories:

1. *Bitmap image* (or raster image): can represented by our image model I(r, c), where we have pixel data and corresponding brightness values stored in file format.

2. *Vector images*: refer to the methods of representing lines, curves shapes by storing only the key points. These key points are sufficient to define the shapes, and the process of turning theses into an image is called rendering. After the image has been rendered, it can be thought of as being in bitmap format where each pixel has specific values associated with it.

The differences between vector and raster graphics are:

1- Raster graphics are composed of pixels, while vector graphics are composed of paths.

2- A raster graphic, such as a gif or jpeg, is an array of pixels of various colors, which together form an image. A vector graphic, such as an .eps file or Adobe Illustrator file, is composed of paths, or lines, that are either straight or curved.

3- Because vector graphics are not made of pixels, the images can be scaled to be very large without losing quality. Raster graphics, on the other hand, become "blocky," since each pixel increases in size, as the image is made larger.

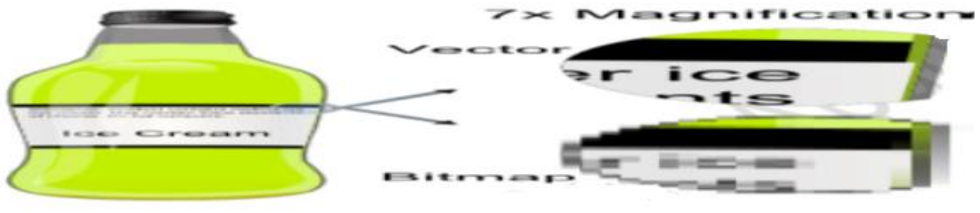


Figure (1.12): vector and bitmap image.

Image file formats are standardized means of organizing and storing digital images. Image files are composed of digital data in one of these formats that can be rasterized for use on a computer display or printer. An image file format may store data in uncompressed, compressed, or vector formats. Once rasterized, an image becomes a grid of pixels, each of which has a number of bits to designate its color equal to the color depth of the device displaying it.

Many image types can be converted to one of other type by easily available image conversion software. Field related to computer imaging is that computer graphics.

The most the type of file format falls into category of bitmap images. In general, these types of images contain both header information and the raw pixel data. The header information contains information regarding:

1. The number of rows (height)
2. The number of columns (Width)
3. The number of bands.
4. The number of bit per pixel.
5. The file type.
6. Additionally, with some of the more complex file formats, the header may contain information about the type of compression used and other necessary parameters to create the image, I(r, c).

1.9.2 Image File Format:

1. BMP format:

It’s a compressed format and the data of image are located in the field of data while there are two fields :

One for header (54 byte) that contains the image information such as (height ,width , no. of bits per pixel, no of bands , the file type).

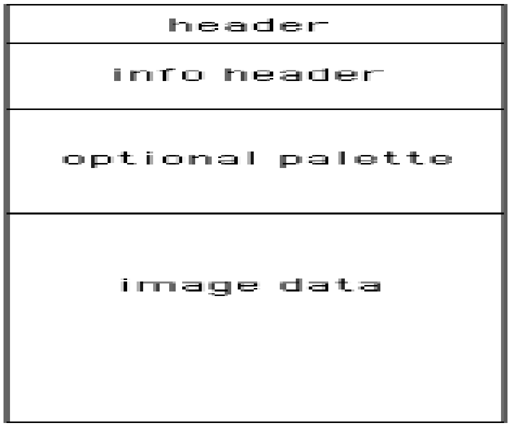
The second field is the color map or color palette for gray level image, where its length is 0-255.

The **BMP file format**, also known as **bitmap image file** or **device independent bitmap (DIB) file format** or simply a **bitmap**, is a raster graphics image file format used to store bitmap digital images, independently of the display device (such as a graphics adapter), especially on Microsoft Windows and OS/2 operating systems.

The BMP file format is capable of storing 2D digital images of arbitrary width, height, and resolution, both monochrome and color, in various color depths, and optionally with data compression, alpha channels, and color profiles. BMP files are an historic (but still commonly used) file format for the operating system called "Windows". BMP images can range from black and white (1 bit per pixel) up to 24-bit color (16.7 million colors). While the images can be compressed, this is rarely used in practice and will not be discussed in detail here.

Structure

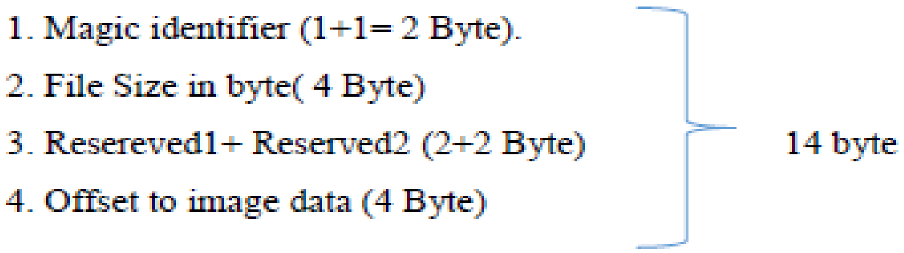
A BMP file consists of either 3 or 4 parts as shown in the following diagram



The first part is a header, this is followed by an information section, if the image is indexed color then the palette follows, and last of all is the pixel data. Information such as the image width and height, the type of compression, the number of colors is contained in the information header.

Header

The header consists of the following fields. Note that we are assuming short int of 2 bytes, int of 4 bytes, and long int of 8 bytes.

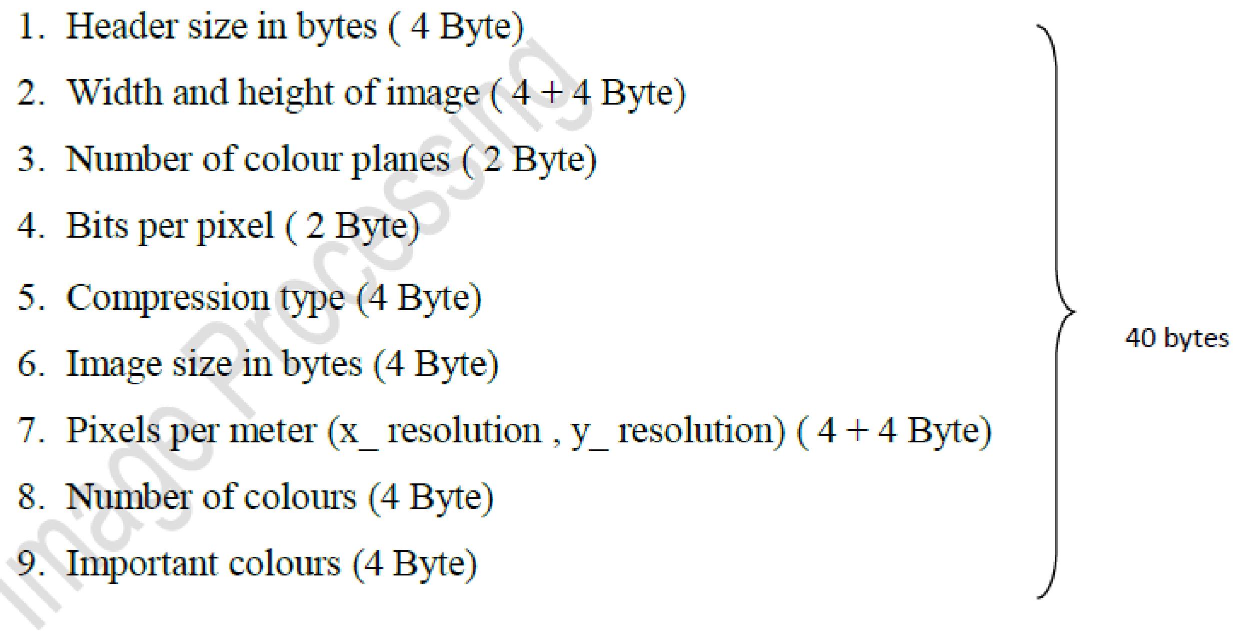


1. Magic identifier (1+1= 2 Byte).
2. File Size in byte(4 Byte)
3. Resereved1+ Reserved2 (2+2 Byte)
4. Offset to image data (4 Byte)

14 byte

Information

The image info data that follows is 40 bytes in length, structure given below. The fields of most interest below are the image width and height, the number of bits per pixel (should be 1, 4, 8 or 24), the number of planes (assumed to be 1 here), and the compression type (assumed to be 0 here).



1. Header size in bytes (4 Byte)
2. Width and height of image (4 + 4 Byte)
3. Number of colour planes (2 Byte)
4. Bits per pixel (2 Byte)
5. Compression type (4 Byte)
6. Image size in bytes (4 Byte)
7. Pixels per meter (x_ resolution , y_ resolution) (4 + 4 Byte)
8. Number of colours (4 Byte)
9. Important colours (4 Byte)

40 bytes

The compression types supported by BMP are listed below:

0 - no compression

1 - 8 bit run length encoding

2 - 4 bit run length encoding

3 - RGB bitmap

2. TIFF (Tagged Image File Format) and GIF (Graphics Interchange Format):

It is one of the most popular and flexible of the current public domain raster file formats. They are used on World Wide Web (WWW). GIF files are limited to a maximum of 8 bits/pixel and allows for a type of compression called LZW. The GIF image header is 13 byte long & contains basic information.

3. JPEG (Joint photo Graphic Experts Group):

This is the right format for those photo images, which must be very small files, for example, for web sites or for email. JPG is often used on digital camera memory cards. The JPG file is wonderfully small, often compressed to perhaps only 1/10 of the size of the original data, which is a good thing when modems are involved. However, this fantastic compression efficiency comes with a high price. JPG uses lossy compression (lossy meaning "with losses to quality"). Lossy means that some image quality is lost when the JPG data is compressed and saved, and this quality can never be recovered. JPEG images compression is being used extensively on the WWW. It’s, flexible, so it can create large files with excellent image equality.

4. VIP (visualization in image processing) formats:

It is developed for the CVIP tools software, when performing temporary images are created that use floating point representation, which is beyond the standard 8-bit/pixel. To represent this type of data the remapping is used, which is the process of taking original image and adding an equation to translate it to the range (0-225).

Chapter Two: Image Analysis

2.1. Image Analysis

Image analysis involves manipulating the image data to determine exactly the information necessary to help solve a computer-imaging problem. This analysis is typically part of a larger process, is iterative in nature and allows us to answer application specific equations: Do we need color information? Do we need to transform the image data into the frequency domain? Do we need to segment the image to find object information? What are the important features of the image?

Image analysis is primarily **data reduction** process. As we have seen, images contain enormous amount of data, typically on the order hundreds of kilobytes or even megabytes. Often much of this information is not necessary to solve a specific computer-imaging problem, so primary part of the image analysis task is to determine exactly what information is necessary. Image analysis is used both computer vision and image processing.

For computer vision, the product is typically the extraction of high-level information for computer analysis or manipulation. This high level information may include shape parameter to control a robotics manipulator or color and texture features to help in diagnosis of a skin tumor.

In image processing application, image analysis methods may be used to help determine the type of processing required and the specific parameters needed for that processing. For example, determine the degradation function for an image restoration procedure, developing an enhancement algorithm and determining exactly what information is visually important for image compression methods.

2.2 System Model

The image analysis process can be broken down into three primary stages:

1. Preprocessing**.** **2.** Data Reduction. **3.** Features Analysis.

1. Preprocessing:

Is used to remove noise and eliminate irrelevant, visually unnecessary information. Preprocessing steps might makes the primary data reduction and analysis task easier. They include operations related to:

- Gray –level or spatial quantization (reducing the number of bits per pixel or the image size).
- Is used to remove noise and eliminate irrelevant, visually unnecessary information, (Noise is unwanted information that can result from the image acquisition process) □ Finding regions of interest for further processing.
- Performing basic algebraic operation on image.
- Enhancing specific image features.
- Reducing data in resolution and brightness.
- Finding regions of interest for further processing.

Preprocessing is a stage where the requirements are typically obvious and simple, such as removal of artifacts from images or eliminating of image information that is not required for the application.

For example, in one application we needed to eliminate borders from the images that have been digitized from film.

Another example of preprocessing step involves a robotics gripper that needs to pick and place an object ; for this we reduce a gray-level image to binary (two-valued) image that contains all the information necessary to discern the object ‘s outlines.

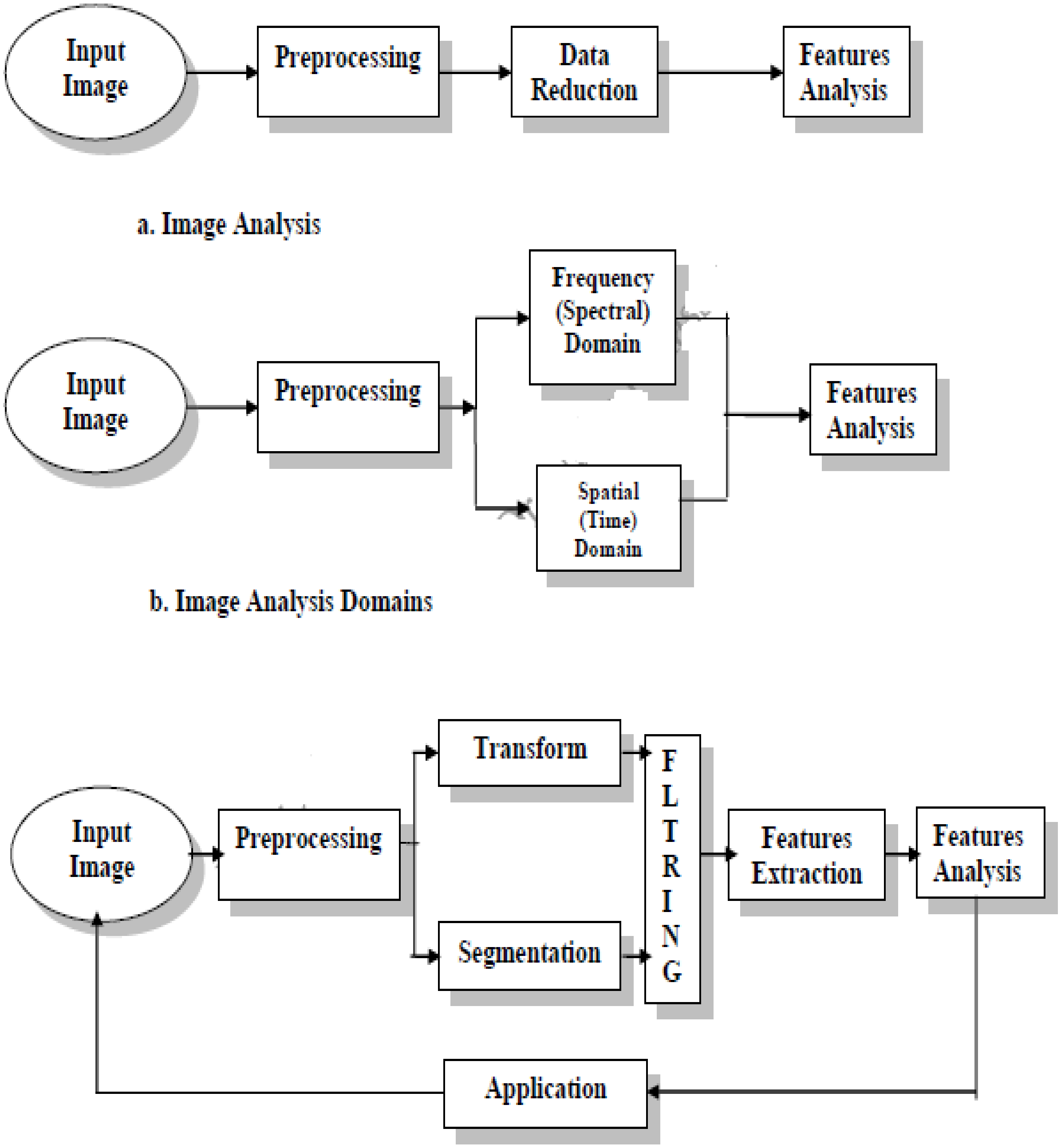
2. Data Reduction:

Involves either reducing the data in the spatial domain or transforming it into another domain called the frequency domain, and then extraction features for the analysis process.

3. Features Analysis:

The features extracted by the data reduction process are examine and evaluated for their use in the application.

After preprocessing, we can perform segmentation on the image in the spatial domain or convert it into the frequency domain via a mathematical transform. After these processes, we may choose to filter the image. This filtering process further reduces the data and allows us to extract the feature that we may require for analysis.



a. Image Analysis

b. Image Analysis Domains

Figure (2.1): System Model of Image Analysis

2.3 <u>Region Of Interest (ROI)</u>

For image analysis, we want to investigate more interested area within the image, called region of interest (ROI). To do this we need operation that modifies the spatial coordinates of the image, and these are categorized as image geometry operations. The image geometry operations discussed here include:

Crop , Zoom, enlarge ,shrink, translate and rotate.

<u>1.</u> **<u>The image crop process</u>** is the process of selecting a small portion of the image, a sub image and cutting it away from the rest of the image. After we have cropped a sub image from the original image we can zoom in on it by <u>enlarge </u>it.

The zoom process can be implemented by the following techniques:

A. Zero-Order Hold.
B. First _Order Hold.
C. Convolution.

<u>A.</u> ***<u>Zero-Order hold</u>*****:** is achieved by repeating previous pixel values, thus creating a blocky effect as in the following figure:

Original Image Array

$$\begin{bmatrix} 8 & 4 & 8 \\ 4 & 8 & 4 \\ 8 & 2 & 8 \end{bmatrix}$$

Image After Applying Zero-Order-Hold

$$\begin{bmatrix} 8 & 8 & 4 & 4 & 8 & 8 \\ 8 & 8 & 4 & 4 & 8 & 8 \\ 4 & 4 & 8 & 8 & 4 & 4 \\ 4 & 4 & 8 & 8 & 4 & 4 \\ 8 & 8 & 2 & 2 & 8 & 8 \\ 8 & 8 & 2 & 2 & 8 & 8 \end{bmatrix}$$

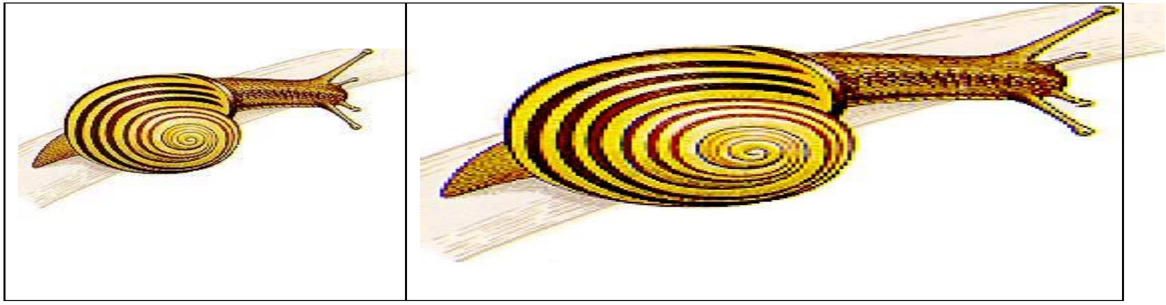


Figure (2.2): Zero _Order Hold Method

B. *First _Order Hold*: is performed by finding linear interpolation between a adjacent pixels, i.e., finding the average value between two pixels and use that as the pixel value between those two, we can do this for the rows first as follows:

Original Image Array

$$\begin{bmatrix} 8 & 4 & 8 \\ 4 & 8 & 4 \\ 8 & 2 & 8 \end{bmatrix}$$

Image with Rows Expanded

$$\begin{bmatrix} 8 & 6 & 4 & 6 & 8 \\ 4 & 6 & 8 & 6 & 4 \\ 8 & 5 & 2 & 5 & 8 \end{bmatrix}$$

The first two pixels in the first row are averaged (8+4)/2=6, and this number is inserted between those two pixels. This is done for every pixel pair in each row.

$$\begin{bmatrix} 8 & 6 & 4 & 6 & 8 \\ 6 & 6 & 6 & 6 & 6 \\ 4 & 6 & 8 & 6 & 4 \\ 6 & 5.5 = 6 & 5 & 5.5 = 6 & 6 \\ 8 & 5 & 2 & 5 & 8 \end{bmatrix}$$

Image with rows and columns expanded

This method allows us to enlarge an N×N sized image to a size of (2N-1) × (2N-1) and be repeated as desired.

Example: Enlarge the following sub image using:

1- First –Order- Hold.

2- Zero-Order _Hold.

$$\begin{bmatrix} 150 & 50 & 30 \\ 40 & 100 & 80 \\ 120 & 60 & 60 \end{bmatrix}$$

Solution:

a. First-Order_Hold.

Size of original image is: 3x3

Size of the resultant image is: 5x5

(150+50)/2=100 (50+30)/2=40 (40+100)/2=70

(100+80)/2=90 (120+60)/2=90 (60+60)/2=60

Image with row expanded

$$\begin{bmatrix} 150 & 100 & 50 & 40 & 30 \\ 40 & 70 & 100 & 90 & 80 \\ 120 & 90 & 60 & 60 & 60 \end{bmatrix}$$

(150+40)/2=95 (100+70)/2=85 (50+100)/2=75

(40+90)/2=66 (30 + 80)/2=55 (40 +120)/2=80

(70 + 90)/2=80 (100 + 60)/2=80 (90 + 60)/2=75 (80 +60)/2=70

Image with row and column expanded

$$\begin{bmatrix} 150 & 100 & 50 & 40 & 30 \\ 95 & 85 & 75 & 66 & 55 \\ 40 & 70 & 100 & 90 & 80 \\ 80 & 80 & 80 & 75 & 70 \\ 120 & 90 & 60 & 60 & 60 \end{bmatrix}$$

b. Zero-Order_Hold

The original image size is 3x3.

After enlargement, the size will be 6x6

$$\begin{bmatrix} 150 & 150 & 50 & 50 & 30 & 30 \\ 150 & 150 & 50 & 50 & 30 & 30 \\ 40 & 40 & 100 & 100 & 80 & 80 \\ 40 & 40 & 100 & 100 & 80 & 80 \\ 120 & 120 & 60 & 60 & 60 & 60 \\ 120 & 120 & 60 & 60 & 60 & 60 \end{bmatrix}$$

B. ***Convolution*:** This process requires a mathematical process to enlarge an image.

This method required two steps:

1. Extend the image by adding rows and columns of zeros between the existing rows and columns.
2. Perform the convolution.
3.

The image is extended as follows**:**

Original Image Array

$$\begin{pmatrix} 3 & 5 & 7 \\ 2 & 7 & 6 \\ 3 & 4 & 9 \end{pmatrix}$$

Image extended with zeros

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 5 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 7 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 4 & 0 & 9 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Next, we use convolution mask, which is slide a cross the extended image, and perform simple arithmetic operation at each pixel location

Convolution mask for first-order hold

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

The convolution process requires us to overlay the mask on the image, multiply the coincident values and sum all these results. This is equivalent to finding the vector inner product of the mask with underlying sub image. The vector inner product is found by overlaying mask on sub image. Multiplying coincident terms, and summing the resulting products.

For example, if we put the mask over the upper-left corner of the image, we obtain (from right to left, and top to bottom):

1/4(0) +1/2(0) +1/4(0) +1/2(0) +1(3) +1/2(0) +1/4(0) +1/2(0) +1/4(0) =3 Note that the existing image values do not change. The next step is to slide the mask over by on pixel and repeat the process, as follows:

1/4(0)+1/2(0) +1/4(0) +1/2(3) +1(0) +1/2(5) +1/4(0) +1/2(0) +1/4(0) =4.

Note this is the average of the two existing neighbors. This process continues until we get to the end of the row, each time placing the result of the operation in the location corresponding to center of the mask. When the end of the row is reached, the mask is moved down one row, and the process is repeated row by row. This procedure has been performed on the entire image, the process of **sliding, multiplying and summing** ***is called convolution***.

In this process, the output image must be put in a separate image array called a buffer, so that the existing values are not overwritten during the convolution process. If we call the convolution mask M(r, c) and the image I(r, c), the convolution equation is given by:

$$\sum_{x=-\infty}^{\infty} \sum_{y=-\infty}^{\infty} I(r-x, c-y) M(x, y)$$

<u>The steps of the convolution process can be summarized as follows:</u>

a. Overlay the convolution mask in the upper-left corner of the image. Multiply coincident

terms, sum, and put the result into the image buffer at the location that corresponds to the masks current center, which is (r,c)=(1,1).

<u>b.</u> Move the mask one pixel to the right , multiply coincident terms sum , and place the new results into the buffer at the location that corresponds to the new center location of the convolution mask which is now at (r,c)=(1,2), continue to the end of the row.

<u>c.</u> Move the mask down one row and repeat the process until the mask is convolved with the entire image. Note that we lose the outer row(s) and columns(s).

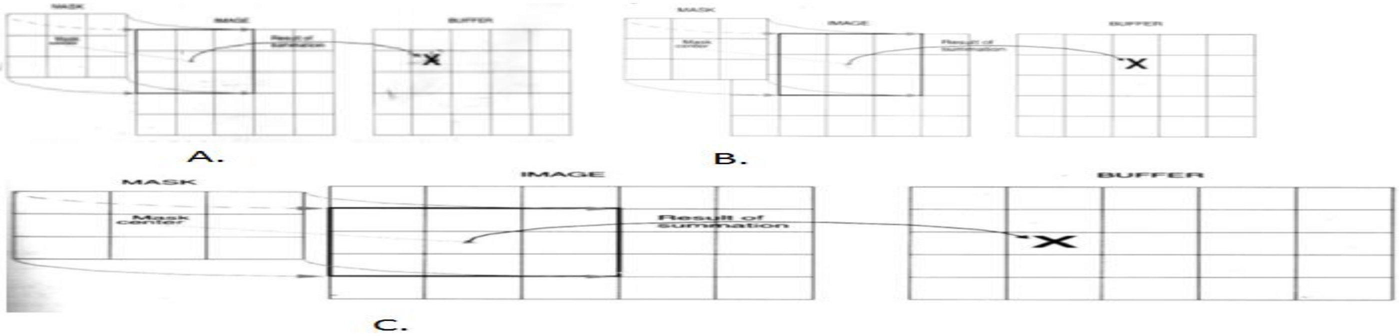


Figure (2.2. a,b and c) the convolution process.

These methods will only allow us to enlarge an image by a factor of (2N-1), but what if we want to enlarge an image by something other than a factor of (2N-1)?

To do this we need to apply a more general method. We take two adjacent values and linearly interpolate more than one value between them.

This is done by define an enlargement number k and then following this process:

1. Subtract the result by k.

2. Divide the result by k.

3. Add the result to the smaller value, and keep adding the result from the second step in a running total until all (k-1) intermediate pixel locations are filled.

Example: Enlarge the following sub image three times its original size using k method.

$$\begin{bmatrix} 130 & 160 & 190 \\ 160 & 190 & 220 \end{bmatrix}$$

Solution:

The size of the result image will be k(N-1)+1,where k=3

So the sub image with size 2x3 will be

3(2-1)+1=4 row and 3(3-1)+1=7 column (4 x 7)

For column:

1: 130-160=30 K=3 then 30/3=10

k-1=3-1=2 no. of column to be inserted

130+10*1=140, 130+10*2=150

2: 160-190=30/3=10 160+10 *1=170, 160+10*2=180

3: 160-190=30/3=10 160+10 *1=170, 160+10*2=180

4: 190-220=30/3=10 190+10 *1=200, 190+10*2=210

The same thing repeated for the row. The resultant image is:

$$\begin{bmatrix} \boxed{130} & 140 & 150 & \boxed{160} & 170 & 180 & \boxed{190} \\ 140 & 150 & 160 & 170 & 180 & 190 & 200 \\ 150 & 160 & 170 & 180 & 190 & 200 & 210 \\ \boxed{160} & 170 & 180 & \boxed{190} & 200 & 210 & \boxed{220} \end{bmatrix}$$

2.3.2 Image Shrinking

The process opposite to enlarge an image is shrinking it. This is not typically done to examine a ROI more closely but to reduce the amount of data that needs to be processed.

Image shrinking is accomplished by taking groups of pixels that are spatially adjacent and mapping them to one pixel.

This can be done in one of ***three* *ways:***

- **Averaging . Median . Decimation.**

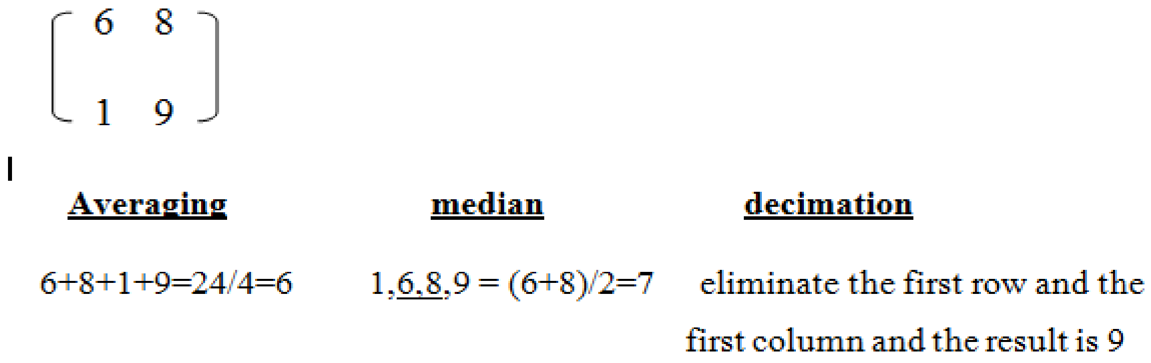
For averaging method, we take all the pixels in each group (for example 2×2 block of pixels) and find the average gray level by summing the values select and divided by 4.

For the second method, **median,** we sort all the pixels values from lowest to highest and then select the middle value.

The third method decimation, also known as sub sampling, entails simply eliminating some of the data. For example to reduce the image by a factor of two, we simply take every other row and column and delete them.

Example

Shrink the following sub image using the three shrinking methods (Averaging, Median, Decimation).



$$\begin{bmatrix} 6 & 8 \\ 1 & 9 \end{bmatrix}$$

Averaging	**median**	**decimation**
6+8+1+9=24/4=6	$1,\underline{6,8},9 = (6+8)/2=7$	eliminate the first row and the first column and the result is 9

Example

Shrink the following sub image using the three shrinking methods (median , decimation)

15	20	25	30	25	20	15
20	21	21	25	21	21	20
25	22	22	20	22	22	25
30	25	20	15	20	25	30

Median :

20 25 21
25 20 25

Decimation

21 25 20
25 15 30

Zoom.

These methods will only allow us to enlarge an image by a factor of (k) times.

Example

if you have a sub image of size 2 ×2 , zoom the sub image to twice (2)times using zero older hold method.

$$I= \begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$$

$$I' = \begin{bmatrix}1 & 1 & 2 & 2\\ 3 & 3 & 4 & 4\end{bmatrix} \quad Row\ wisng\ e\ zooming$$

$$\mathbf{I''}= \begin{bmatrix}1 & 1 & 2 & 2\\ 1 & 1 & 2 & 2\\ 3 & 3 & 4 & 4\\ 3 & 3 & 4 & 4\end{bmatrix} \quad column\ wisng\ e\ zooming$$

Example

Enlarge the following sub image to three times its original size using the general method of zooming techniques. $I = \begin{bmatrix}15 & 30 & 15\\ 30 & 15 & 30\end{bmatrix}_{2\times3}$

Sol:

K=3, K-1=2

Row wise zooming:

- Take the first two adjacent pixels. Which are 15 and 30.
- Subtract 15 from 30: 30-15 = 15.
- Divide 15 by k: 15/k = 15/3 = **5** ------→ **OP**
- Add **OP** **(5)** to the lower number: 15 + 5 = 20.
- Add **OP** (5) to 20 again. : 20 + 5 = 25.

We do that 2 times because we have to insert **(k-1)** values.

Now repeat this step for the next two adjacent pixels. It is shown in the first table. After inserting the values , you have to sort the inserted values in ascending order, so there remains a symmetry between them. It is shown in table 2

Table 1:

15	20	25	30	20	25	15
30	20	25	15	20	25	30

Table 2

15	20	25	30	**25**	**20**	15
30	**25**	**20**	15	20	25	30

<u>Column wise zooming :</u>

The same procedure has to be performed column wise. The procedure include taking the two adjacent pixel values, and then subtracting the smaller from the bigger one. Then after that , you have to divide it by k. Store the result as OP. Add OP to smaller one, and then again add OP to the value that comes in first addition of OP. Insert the new values;

The final output is :

Table 3

15	20	25	30	25	20	15
20	21	21	25	21	21	20
25	22	22	20	22	22	25
30	25	20	15	20	25	30

2.4. <u>Image Algebra</u>

There are two primary categories of algebraic operations applied to image:

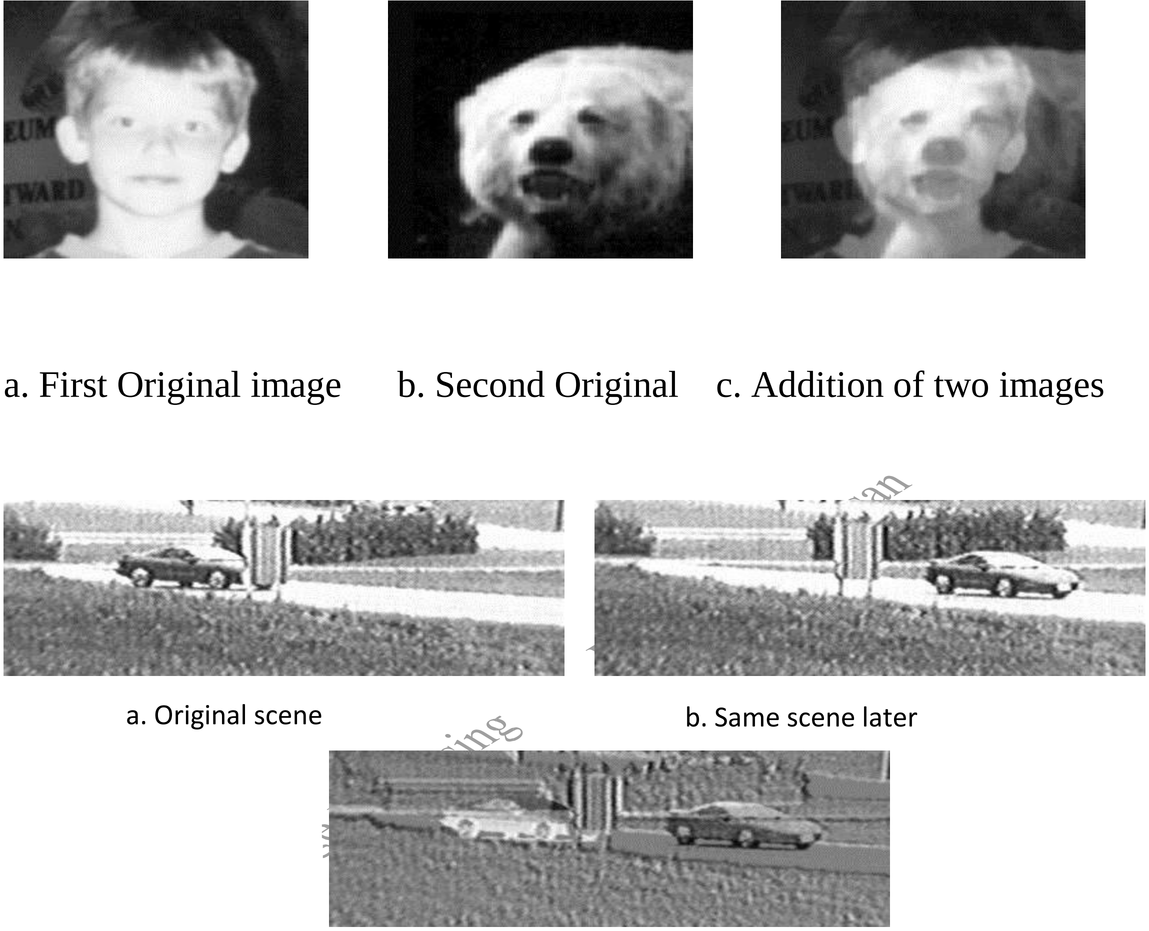
1. **Arithmetic operations.(**Addition,subtraction,divisionand multiplications)
2. **Logic operations. (**AND, OR and NOT)

These operations which require only one image, and are done on a pixel–by-pixel basis.

To apply the arithmetic operations on two images, we simply operate on corresponding pixel values, which means that the value of a pixel in the output image depends only on the values of the corresponding pixels in the input images. Hence, the images normally have to be of the same size. For example to add image I_1 and I_2 to create I_3:

$$\overset{I_1}{\begin{pmatrix}3&4&7\\3&4&5\\2&4&6\end{pmatrix}} + \overset{I_2}{\begin{pmatrix}6&6&6\\4&2&6\\3&5&5\end{pmatrix}} = \overset{I_3}{\begin{pmatrix}3+6&4+6&7+6\\3+4&4+2&5+6\\2+3&4+5&6+5\end{pmatrix}} = \begin{pmatrix}9&10&13\\7&6&11\\5&9&11\end{pmatrix}$$

- **<u>Addition</u>** is used to combine the information in two images as shown in figure (2.5). Or adding a constant value (scalar) to an image causes an increase (or decrease if the value is less than zero) in its overall brightness. Applications include development of image restoration algorithm for molding additive noise, and special effects, such as image morphing in motion pictures.
- **<u>Subtraction</u>** of two images is often used to **detect motion**, consider the case where nothing has changed in a sense; the image resulting from subtraction of two sequential image is filled with zero-a black image. If something has moved in the scene, subtraction produces a nonzero result at the location of movement. Subtraction process also used to detect the defects in the images. Applications include Object tracking , Medical imaging. Subtraction can result in a negative values for certain pixels. When this occurs with unsigned data types, such as uint8 or uint16, the imsubtract function truncates the negative value to zero (0), which displays as black.
- ***<u>Multiplication and Division</u>*** are used **to adjust the brightness of an image**. (The Brightness adjustment is often used as a processing step in image enhancement). **Multiplication of the pixel values by a (scalar) smaller than one will darken the image, while multiplication by scalar greater than on will brightness the image**. Multiplication process often used for masking operations, see figure (2.7).

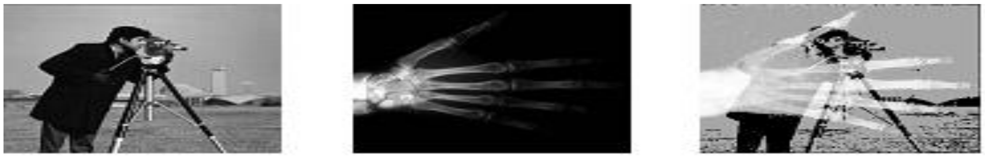


a. First Original image b. Second Original c. Addition of two images

a. Original scene b. Same scene later

c. Subtraction of scene a from scene b

Figure (2.5): Image Addition, Image Subtraction.

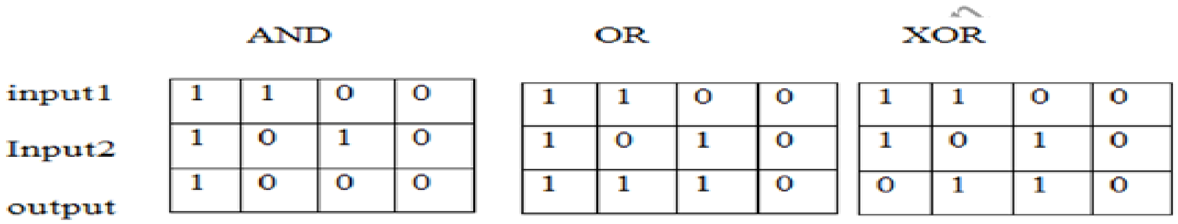


a. Cameraman image b.X-ray image of hand c. Multiplication of two images

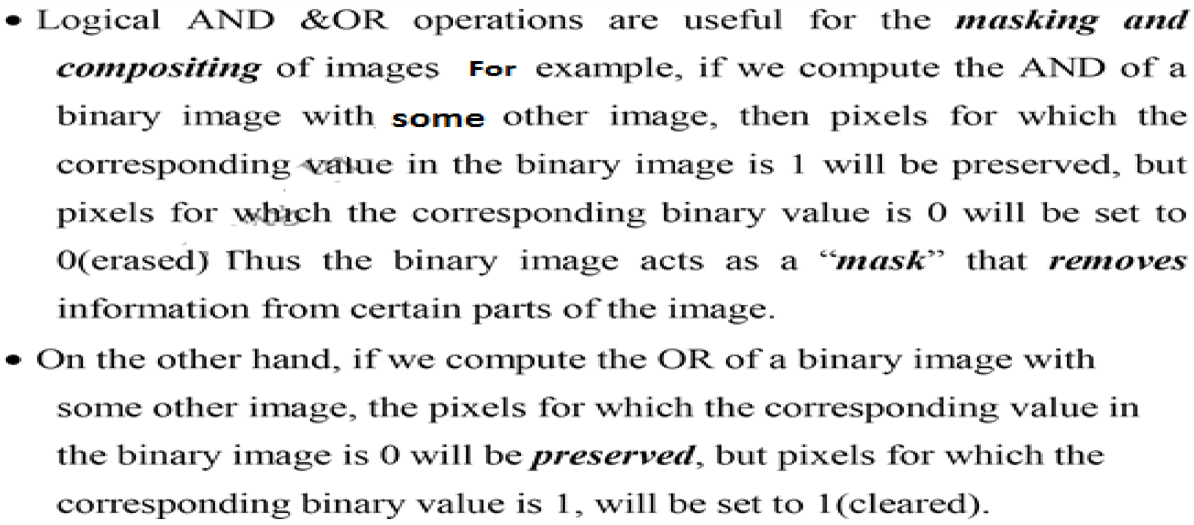
Figure (2.6) Image Multiplication.

Figure (2.7) Image Division.

Logical operations apply only ***to binary images***, whereas arithmetic operations apply to multi-valued pixels. Logical operations are basic tools in binary image processing, where they are used for tasks such as ***masking***, ***feature detection***, and ***shape analysis***. Logical operations on entire image are performed pixel–by–pixel. Because the AND operation of two binary variablesis1only when both variables are 1, the result at any location in a resulting AND image is1 only if the corresponding pixels in the two input images are 1. As logical operation involve only one pixel location at a time, they can be done in place, as in the case of arithmetic operations .The XOR (exclusive OR) operation yields a 1 when one or other pixel (but not both) is 1,and it yields a 0 otherwise. The operation is unlike the OR operation, whichis1, when one or the other pixel is1,or both pixels are 1.



	AND				OR				XOR			
input1	1	1	0	0	1	1	0	0	1	1	0	0
Input2	1	0	1	0	1	0	1	0	1	0	1	0
output	1	0	0	0	1	1	1	0	0	1	1	0



- Logical AND &OR operations are useful for the ***masking and compositing*** of images For example, if we compute the AND of a binary image with some other image, then pixels for which the corresponding value in the binary image is 1 will be preserved, but pixels for which the corresponding binary value is 0 will be set to 0(erased) Thus the binary image acts as a "***mask***" that ***removes*** information from certain parts of the image.
- On the other hand, if we compute the OR of a binary image with some other image, the pixels for which the corresponding value in the binary image is 0 will be ***preserved***, but pixels for which the corresponding binary value is 1, will be set to 1(cleared).

So, masking is a simple method to extract a region of interest (ROI) from an image.

Example: A white square ANDed with an image will allow only the portion of the image coincident with the square to appear in the output image with the background turned black; and a black square ORd with an image will allow only the part of the image corresponding to the black square to appear in the output image but will turn the rest of the image white. This process is called ***image masking*** as shown in figure (2.8).

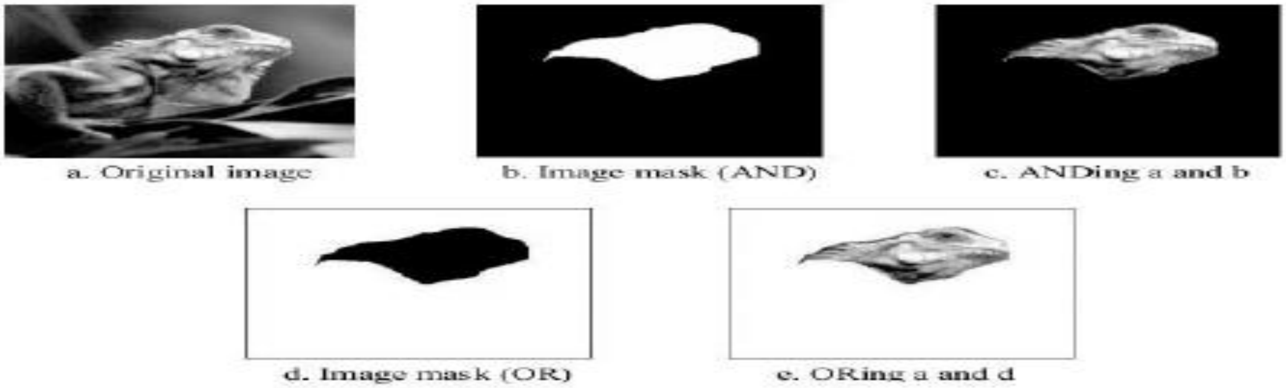

a. Original image
b. Image mask (AND)
c. ANDing a and b
d. Image mask (OR)
e. ORing a and d

Figure (2.8) Image Masking.

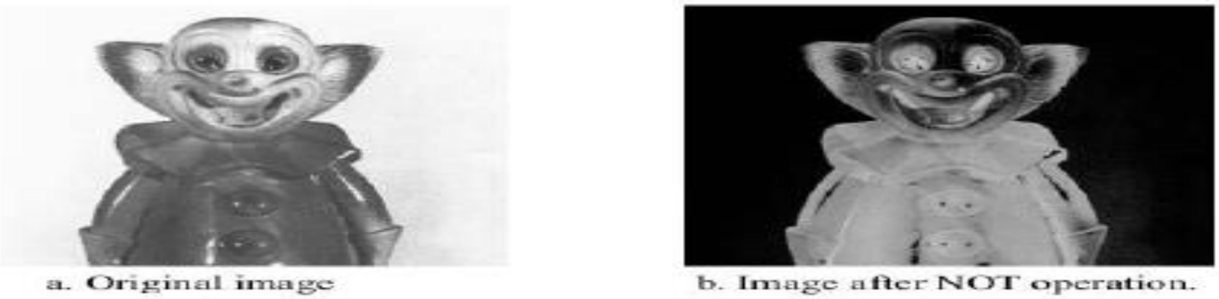

a. Original image
b. Image after NOT operation.

Figure (2.9) Complement Image.

In addition to masking, logical operation can be used in feature detection. Logical operation can be used to compare between two images.

1- AND

This operation can be used to find the ***similarity*** white regions of two different images (it required two images).

g (x,y) = a (x,y) ^ b (x,y)

Example: A logic AND is performed on two images, suppose the two corresponding pixel values are $\mathbf{(111)_{10}}$ is one image and $\mathbf{(88)_{10}}$ in the second image. The corresponding bit strings are:

$$\begin{array}{lcll} \mathbf{(111)_{10}} & \longrightarrow & \mathbf{01101111_2} & \\ & & & \mathbf{AND} \\ \mathbf{(88)_{10}} & \longrightarrow & \mathbf{01011000_2} & \\ \hline \mathbf{(72)_{10}} & & \mathbf{01001000} & \end{array}$$

2- Exclusive OR

This operator can be used to find the ***differences*** between white regions of two different images(it requires two images).

$$g(x,y) = a(x,y) \otimes b(x,y)$$

3- NOT

This operator can be performed on grey-level images, it's applied on only one image, and the result of this operation is the ***negative*** of the original image.

The general form is : g(x,y) = maxcolor- f (x,y)

Then

g(x,y) = 255- f (x,y) if the image is 256 color

g(x,y) = 7- f (x,y) if the image is 8 color

and so on.

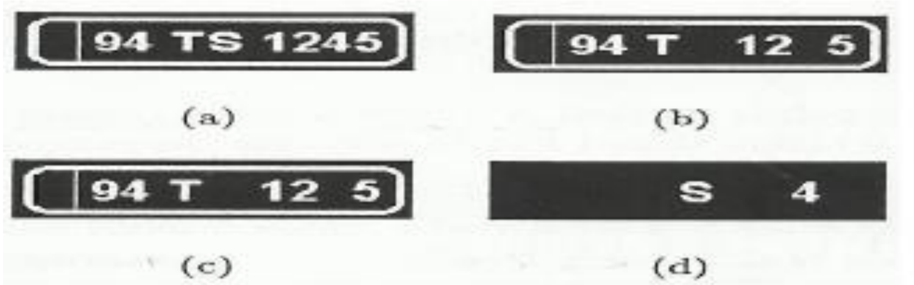


Figure 2.10: a) input image1 a(x,y)
b) input image2 b(x,y)
c) a(x,y) ^ b(x,y)
d) a(x,y)$\otimes$ b(x,y)

Example: Find the output image for the AND logical operation between the following sub images:

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{50}\ _{\mathbf{10}} & \mathbf{11_{10}} \\ \mathbf{18_{10}} & \mathbf{22_{10}} \end{bmatrix},\ \boldsymbol{B} = \begin{bmatrix} \mathbf{10}\ _{\mathbf{10}} & \mathbf{60_{10}} \\ \mathbf{33_{10}} & \mathbf{130_{10}} \end{bmatrix}$$

<u>A</u>

$(50)_{10} = (00110010)_2$, $(11)_{10} = (00001011)_2$

$(18)_{10} = (00010010)_2$, $(22)_{10} = (00010110)_2$

<u>B</u>

$(10)_{10} = (00001010)_2$, $(60)_{10} = (00111100)_2$

$(33)_{10} = (00100001)_2$, $(130)_{10} = (10000010)_2$

A AND B = $\begin{bmatrix} \mathbf{2}\ _{\mathbf{10}} & \mathbf{8_{10}} \\ \mathbf{0_{10}} & \mathbf{2_{10}} \end{bmatrix}$

2.5. Image Quantization

Image quantization is the process of reducing the image data by removing some of the detail information by mapping group of data points to a single point. This can be done by:

1. **Gray_Level reduction (reduce pixel values I(r, c).**
2. **Spatial reduction (reduce the spatial coordinate (r, c).**

The simplest method of gray-level reduction is **Thresholding.**

The Procedure:

-We select a threshold gray _level

- and set everything above that value equal to “1” and everything below the threshold equal to “0”.

NOTE: This effectively turns a gray_level image into a binary (two level) image

Application: is often used as a preprocessing step in the extraction of object features, such as shape, area, or perimeter. Also, the gray _level reduction is the process of taking the data and reducing the number of bits per pixel. **This can be done very efficiency by masking the lower bits via an AND operation. Within this method, the numbers of bits that are masked determine the number of gray levels available**.

Example: We want to reduce 8_bit information containing 256 possible gray_level values down to 32 gray levels possible values.

This can be done by applying the (AND) logical operation for each 8-bit value with the bit string **1111000.** this is equivalent to dividing by eight(2^3), corresponding to the lower three bits that we are masking and then shifting the result left three times. [Gray _level in the image 0-7 are mapped to 0, gray_level in the range 8-15 are mapped to 8 and so on].

We can see that by masking the lower three bits we reduce 256 gray levels to 32 gray levels:

256 ÷ 8= 32

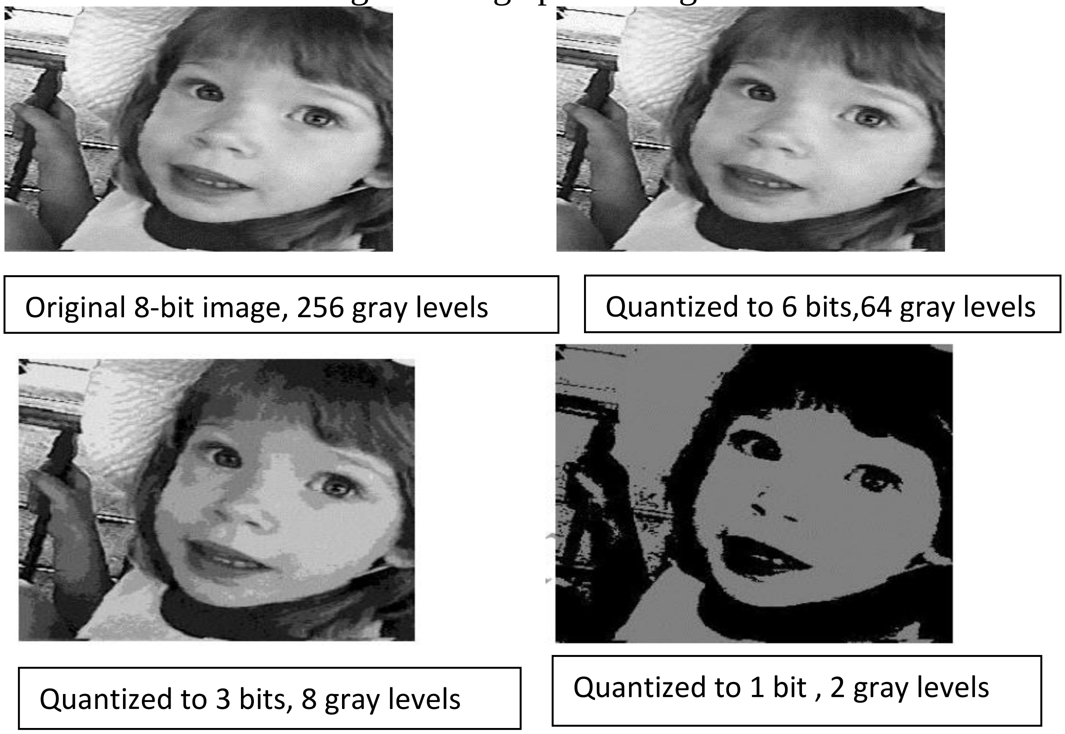
The general case requires us to mask k bits, where 2^k is divided into the original gray-level range to get the quantized range desired. Using this method, we can reduce the number of gray levels to any power of 2: 2,4,6,8, 16, 32, 64 or 128.

- Image quantization by masking to 128 gray levels, this can be done by ANDing each 8-bit value with bit string 11111110(2^1).

- Image quantization by masking to 64 gray_level. This can be done by ANDing each 8-bit value with bit string 11111100(2^2).

As the number of gray levels decreases, we can see increase in a phenomenon called contouring.

Contouring appears in the image as false edges, or lines as a result of the gray _level quantization meth~~o~~d.



Original 8-bit image, 256 gray levels

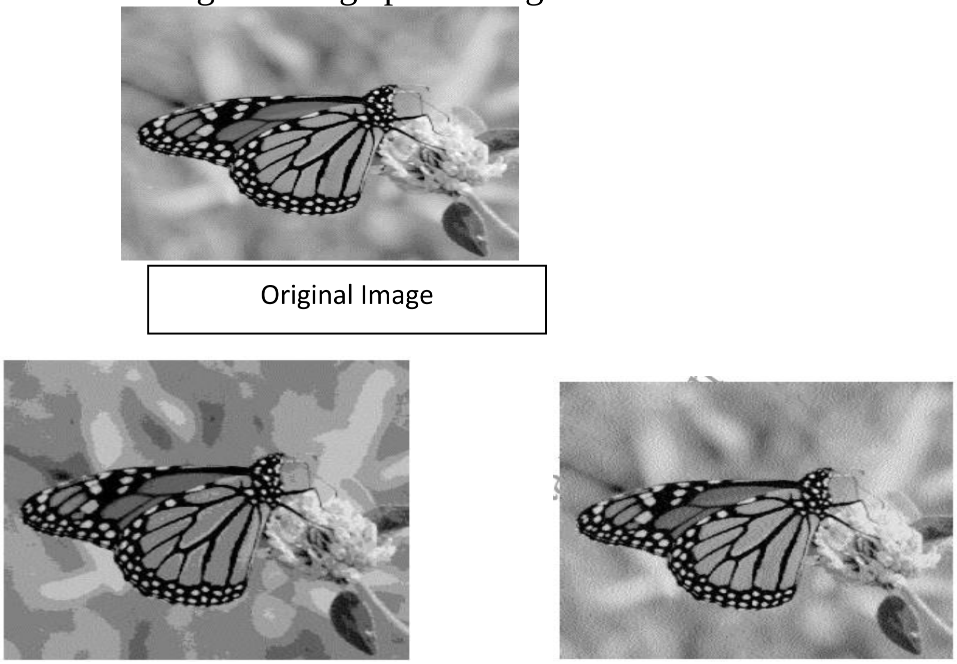
Quantized to 6 bits,64 gray levels

Quantized to 3 bits, 8 gray levels

Quantized to 1 bit , 2 gray levels

Figure (2-17): False Contouring

The **false contouring** effect can be visually improved upon by using an **IGS (improved gray-scale) quantization method**. In this method (IGS) the improvement will be by adding a small random number to each pixel before quantization, which results in a more visually pleasing appearance.



Original Image

Uniform quantization
to 8 levels (3 bits)

IGS quantization
to 8 levels (3 bits)

Figure (2-18): IGS quantization

2.6. Edge Detection

Detecting edges is a basic operation in image processing. The edges of items in an image hold much of the information in the image.

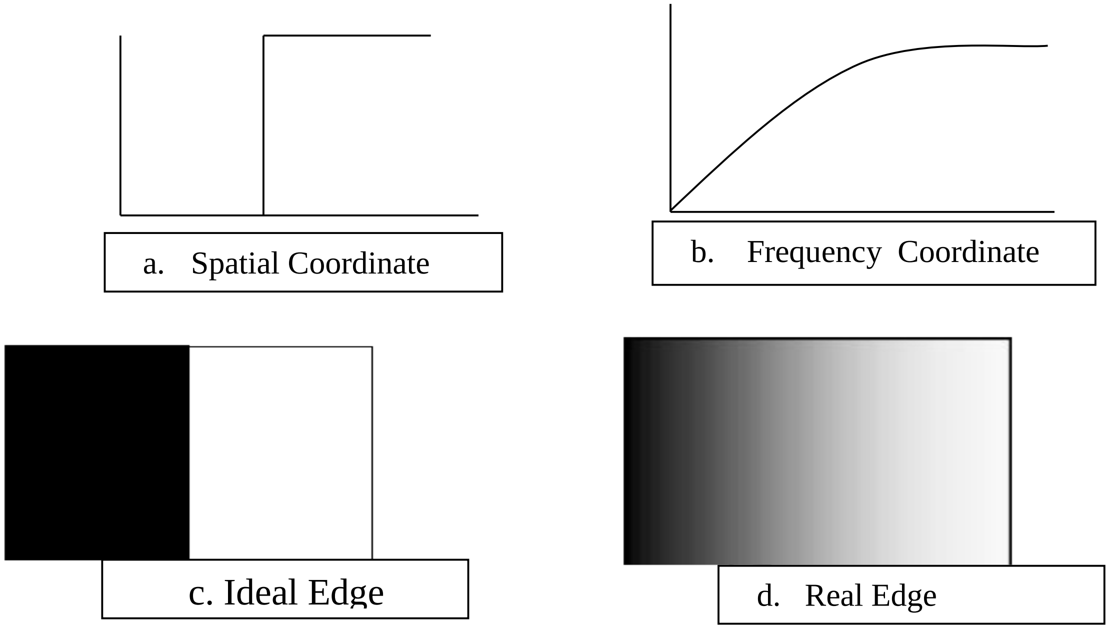
The edges tell you where:

- Items are.
- Their size.
- Shape.
- Something about their texture.

In general, an edge is defined by a discontinuity in gray-level values. Ideally, an edge separates two distinct objects. **In practice, edges are caused by:**

- Change in color or texture or
- Specific lighting conditions present during the image acquisition process.

The following figure illustrates the difference between an ideal edge and a real edge.



a. Spatial Coordinate

b. Frequency Coordinate

c. Ideal Edge

d. Real Edge

Figure (2-19): Ideal vs. Real Edge.

The vertical axis represents brightness, and the horizontal axis shows the spatial coordinates.

The rapid change in brightness characterizes an ideal edge. In the figure (b) we see the representation of real edge, which change gradually. This gradual change is a minor form

of blurring caused by:

- Imaging devices
- The lenses
- Or the lighting and it is typical for real world (as opposed to computer_generated) images.

2.6.1. Edge Detection masks

1- **Sobel Operator:** The Sobel edge detection masks look for edges in both the horizontal and vertical directions and then combine this information into a single metric. These two masks are as follows:

Row Mask **Column Mask**

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

The Procedure:

- These masks are each convolved with the image.
- At each pixel location we now have two numbers: **S1**, corresponding to the result form the row mask and **S2** from the column mask.

- Use S1, S2 to compute two matrices, **the edge magnitude and the edge direction (angle of orientation of the edge),**

which are defined as follows:

Edge Magnitude (EM) = $\sqrt{{S_1}^2 + {S_2}^2}$

Edge Direction (ED) = Tan $^{-1}\left[\frac{S_1}{S_2}\right]$

Example: determine the edges of the following sub image using Sobel edge detector.

$$\begin{bmatrix} 1 & 2 & 2 & 1 \\ 1 & 1 & 0 & 3 \\ 2 & 4 & 1 & 5 \\ 2 & 1 & 2 & 0 \end{bmatrix}$$

Row Mask **Column Mask**

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

Solution:

S1= (2+8+1) – (1+4+2) = 4, **S2=(2+0+1) - (1+2+2) = -2,**

EM (1,1)=$\sqrt{4^2 + (-2)^2}$ **=4.47** **EM(1,2)=**

EM(2,1)= **EM(2,2)=** **…….**

2- Prewitt Operator: The Prewitt is similar to the Sobel but with different mask coefficients. The masks are defined as follows:

Row Mask **Column Mask**

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

The Procedure:

- These masks are each convolved with the image.
- At each pixel location we find two numbers: **P1** corresponding to the result from the row mask and **P2** from the column mask.
- Use P1, P2 to **determine two metrics, the edge magnitude and edge direction (angle of orientation of the edge),** which are defined as follows:

Edge Magnitude = $\sqrt{{P_1}^2 + {P_2}^2}$

Edge Direction = Tan $^{-1}$ $\left[\frac{P_1}{P_2}\right]$

3- Kirsch Compass Mask: the Kirsch edge detection masks are called compass mask s because they are defined by taking a single mask and rotating it to the eight major compass orientations (اتجاهات):

North, north-east, east, south-east, south, south-west, and west and northwest edges in an image. The masks are defined as follows:

-3 -3 5 -3 0 5 -3 -3 5	-3 5 5 -3 0 5 -3 -3 -3	5 5 5 -3 0 -3 -3 -3 -3	5 5 -3 5 0 -3 -3 -3 -3
K0	K1	K2	K3
5 -3 -3 5 0 -3 5 -3 -3	-3 -3 -3 5 0 -3 5 5 -3	-3 -3 -3 -3 0 -3 5 5 5	-3 -3 -3 -3 0 5 -3 5 5
K4	K5	K6	K7

The edge magnitude is defined as the maximum value found by the convolution of each of the mask, with the image.

[Given a pixel, there are eight directions you can travel to a neighboring pixel (above, below , left ,right ,upper left, upper right, lower left, lower right). Therefore there are eight possible directions for an edge. The directional edge detectors can detect an edge in only one of the eight directions. If you want to detect only left to right edges, you would use only one of eight masks. If; however you want to detect all of the edges, you would need to perform convolution over an image eight times using each of the eight masks.

4-Robinson Compass Masks: the Robinson compass masks are used in a manner similar to the Kirsch masks but are easier to implement because they rely only on coefficient of 0, 1 and 2, and are symmetrical about their directional axis-the axis with the zeros, we only need to compute the result on four of the mask, the results. From the other four can be obtained by negating the results from the first four.

The masks are as follows:

$$\underset{\mathbf{R_0}}{\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}} \quad \underset{\mathbf{R_1}}{\begin{pmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{pmatrix}} \quad \underset{\mathbf{R_2}}{\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}} \quad \underset{\mathbf{R_3}}{\begin{pmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{pmatrix}}$$

$$\underset{\mathbf{R_4}}{\begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}} \quad \underset{\mathbf{R_5}}{\begin{pmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{pmatrix}} \quad \underset{\mathbf{R_6}}{\begin{pmatrix} -1 & -2 & -1 \\ \mathbf{0} & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}} \quad \underset{\mathbf{R_7}}{\begin{pmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix}}$$

The edge magnitude is defined as the maximum value found by the convolution of each of the masks with the-image. The edge detection is defined by the mask that produces the maximum magnitude.

It's interesting to note that masks R_0 and $R_7$are the same as the Sobel masks. We can see that any of the edge detection masks can be extended by rotating them in a manner like these compass masks which allow us to extract explicit information about edge in any direction.

4- **Laplacian Operators:** the Laplacian operator described here are similar to the ones used for pre-processing (as described in enhancement filter). The three Laplacian masks that follow represent different approximation of the Laplacian masks are rationally symmetric, which means edges at all orientation contribute to the result. They are applied by selecting one mask and convolving it with the image selecting one mask and convolving it with the image.

Laplacian masks

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{pmatrix} \quad \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

These masks differ from the Laplacian type previously described in that the centre coefficients have been decreased by one. So, if we are only interested in edge information, the sum of the coefficients should be **zero**. If we want to retain most of the information the coefficient should sum to a number greater than zero. Consider an extreme example in which the centre coefficient value will depend most heavily up on the current value, with only minimal contribution from surrounding pixel values.

6- Other Edge Detection Methods

Edge detection can also be performed by *subtraction.* Two methods that use subtraction to detect the edge are **Homogeneity operator** and **Difference operator**.

The workflow of **homogeneity operator** is illustrated as follows:

- subtracts each of the pixels next to the center of the n×n area (where n is usually 3) from the center pixel.
- The result is the maximum of the absolute value of these subtractions. Subtraction in a homogenous region produces zero and indicates an absence of edges.

- A high maximum of the subtractions indicates an edge. This is a quick operator since it performs only subtraction- eight operations per pixel and no multiplication.
- This operator then requires thresholding.

If there is no thresholding then the resulting image looks like a faded copy of the original. Generally thresholding at 30 to 50 gives good result. The thresholding can be varied depending upon the extent of edge detection desired.

The workflow of **difference operator** based on differentiation through:

- calculating the differences between the pixels that surround the centre pixel of an n×n area.
- This operator finds the absolute value of the difference between the opposite pixels, the upper left minus the lower right, upper right minus the lower left, left minus right, and top minus bottom. The result is the maximum absolute value. as in the homogeneity case,
- This operator requires thresholding. However, it is quicker than the homogeneity operator since it uses four integer subtractions as against eight subtractions in homogeneity operator per pixel.

<u>Example</u> :Shown below is how the two operators detect the edge:

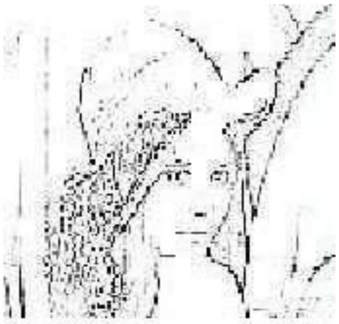
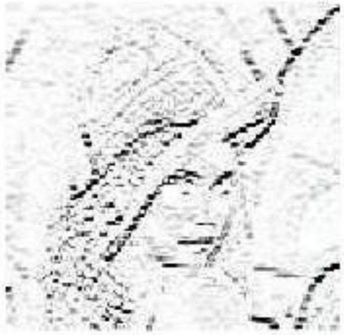
Consider an image block with center pixel intensity 5,

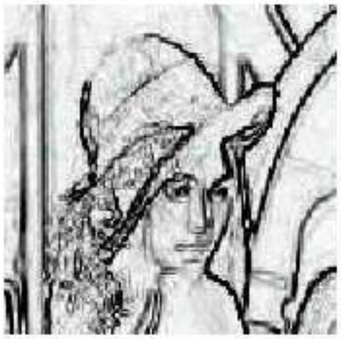
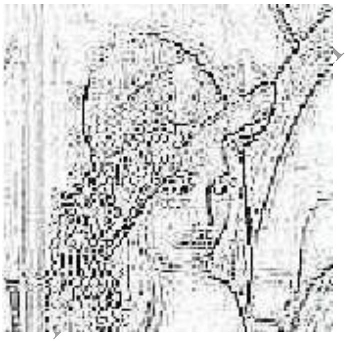
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Output of *homogeneity operator* is:

Max of {| 5-1 |, | 5-2 |, | 5-3 |, | 5-4 |, | 5-6 |, | 5-7 |, | 5-8 |, | 5-9 | } = 4

Output of *difference operator* is: **Max of {| 1-9 |, | 7-3 |, | 4-6 |, | 2-8 | } = 8**

<table>
<tr><td> </td><td></td><td></td></tr>
<tr><td>Original Image</td><td>Edge detect by Sobel' horizontal mask</td><td>Edge detect by Sobel overall mask</td></tr>
</table>

<table>
<tr><td></td><td></td></tr>
<tr><td>Edge detect by Gaussian mask</td><td>Edge detect by Prewitt mask</td></tr>
</table>

Example of Edge Operators

Chapter Threee: Image Restoration

3.1 Introduction

Image restoration methods are used to improve the appearance of an image by application of a restoration process that use **mathematical model** for image degradation.

Example of the type of degradation:

1. Blurring caused by motion or atmospheric disturbance.
2. Geometrics distortion caused by imperfect lenses.
3. Superimposed interface patterns caused by mechanical systems.
4. Noise from electronic source.

It is assumed that the degradation model is known or can be estimated. The idea is to model the degradation process and then apply the inverse process to restore the original image.

3.2. Noise

Noise is any undesired information that contaminates an image. Noise appears in image from a variety of source. The digital image a acquisition process, which converts an optical image into a continuous electrical signal that is then sampled is the primary process by which noise appears in digital images.

At every step in the process there are fluctuations (تذبذب) caused by natural phenomena (ظاهره) that add a random value to exact brightness value for a given pixel. In typical image the noise can be modeled with one of the following distribution:

1. Gaussian (“normal”) distribution.

2. Uniform distribution.

3. Salt _and _pepper distribution.

3.2 Noise Removal using Spatial Filters:

Spatial filtering is typically done for:

1. Remove various types of noise in digital images.

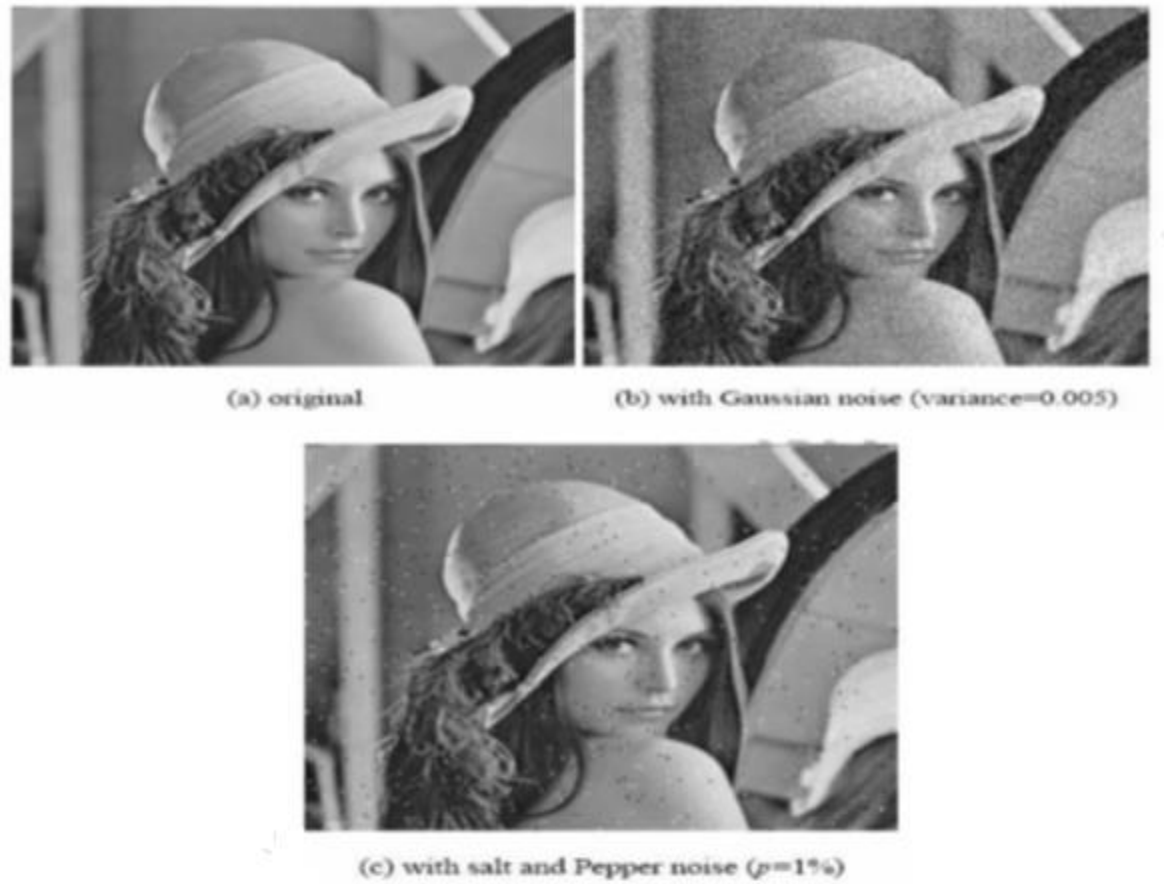
2. Perform some type of image enhancement.

These filters are called spatial filter to distinguish them from frequency domain filter and typically operate on small neighborhood s, 3x3 to 11x11, and some can be implemented as convolution masks. The types of filters are:

1. Mean filters.

2. Order filters (Median filer).

3. Enhancement filters.



(a) original

(b) with Gaussian noise (variance=0.005)

(c) with salt and Pepper noise (p=1%)

Figure (3.1) Image noise

Mean and median filters are used primarily to conceal or remove noise, although they may also be used for special applications. For instance, a mean filter adds “softer” look to an image.

The enhancement filters high lights edges and details within the image. Many Spatial filters are implemented with convolution masks. Because convolution mask operation provides a result that is weighted sum of the values of a pixel and its neighbors, it is called a linear filter.

Overall effects the convolution mask can be predicated based on the general pattern. For example:

- If the coefficients of the mask sum to one, the average brightness of the image will be retained.
- If the coefficients of the mask sum to zero, the average brightness will be lost and will return a dark image.
- If the coefficients of the mask are alternatively positive and negative, the mask is a filter that returns edge information only.
- If the coefficients of the mask are all positive, it is a filter that will blur the image.

3.2.1 mean filters

They are essentially *averaging filter*s. They operate on local groups of pixels called neighborhoods and replace the centre pixel with an average of the pixels in this neighborhood. This replacement is done with a convolution mask such as the following 3X3 mask:

$$\begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix}$$

Arithmetic mean filter smoothing or low-pass filter.

Note that the coefficient of this mask sum to one, so the image brightness will be retained, and the coefficients are all positive, so it will tend to *blur the image* . This type of mean filter smooth out local variations within an image, so it essentially a low pass filter. So a low filter can be used to attenuate image noise that is composed primarily of high frequencies components.

Mean filter ***works well with Gaussian or Uniform noise***, and has the disadvantage of blurring the image edge, or details.

a. Original image b. Mean filtered image

Figure (3.2): Mean Filter.

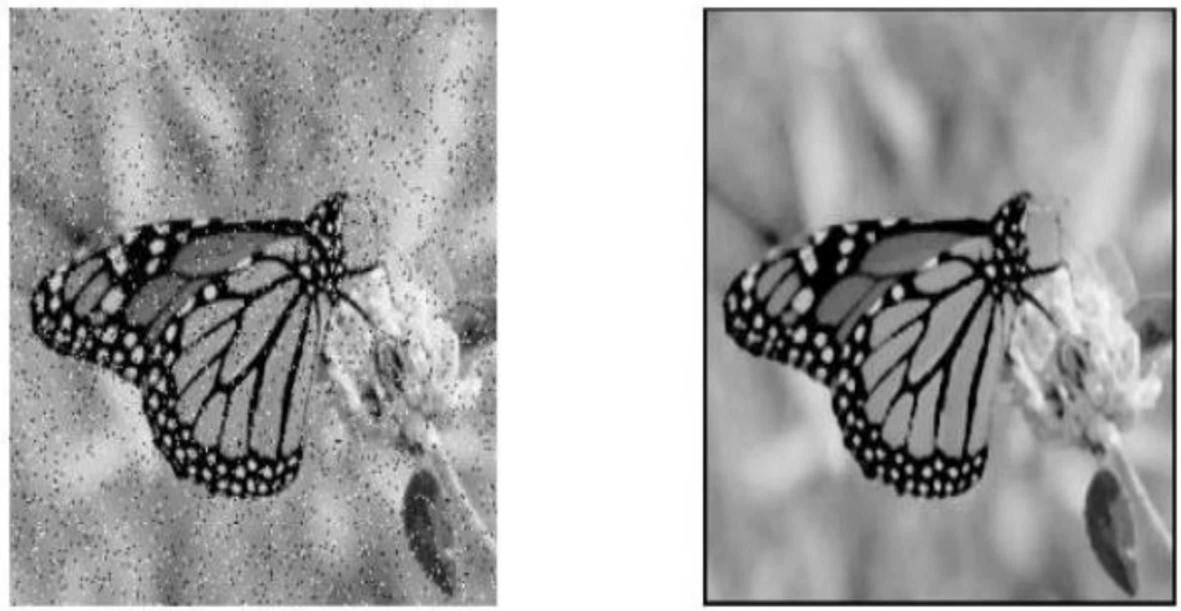
<u>3.2.2 median filter.</u>

It is a non linear filter (order filter). These filters are based on as specific type of image statistics called order statistics. Typically, these filters operate on small sub image, “Window”, and replace the centre pixel value (similar to the convolution process). Order statistics is a technique that arranges the entire pixel in sequential order, given an NXN window (W) the pixel values can be ordered from smallest to the largest.

I1 ≤ **I2** ≤ I**3...………………….≤ IN2**

Where I1**, I2, I3...……, I N2** are the intensity values of the subset of pixels in the image.

Median filters are quite popular because, for certain types of random noise, they provide excellent noise reduction capabilities with considerably less blurring than linear smoothing filters of similar size. Median filter are particularly effective in the presence of its appearance as white and black dots super imposed on an image, i.e. median filter **works best with Salt and Pepper noise.**



a. Salt and pepper noise

b. Median filtered image (3x3)

Figure (3.2) Median filter

Example 3.1

Given the following 3x3 sub image:

$$\begin{bmatrix} 5 & 5 & 6 \\ 3 & 4 & 5 \\ 3 & 4 & 7 \end{bmatrix}$$

We first sort the value in order of size as (3,3,4,4,5,5,5,6,7); the we select the middle value, in this case it is **5** because there are four values above it and four values below it . This 5 is then placed in centre location. A median filter can use a neighborhood of any size, but 3*3, 5*5 and 7*7 are typical.

The median filtering operation is performed on an image by applying the sliding window concepts, similar to what is done with convolution.

The outer rows and columns of an image are not replaced. In practice this is usually not a problem due to the fact that the images are much larger than the masks. And these “wasted” rows and columns are often filled with zeros (or cropped off the image) as mentioned before.

3.2.3. The maximum filter:

It is a Non linear order filter and works by selects the **largest value** within an ordered window of pixels values and replaces the central pixel with the ***lightest one*** . The maximum filters work best for pepper-type noise (low values).

3.2.4 The minimum filter: It is a Non linear order filter and works by selects the smallest value within an ordered window of pixels values and replaces the central pixel with the **darkest one** in the ordered window .

The minimum filters works best for salt- type noise (high values).

3.2.6 The Enhancement filter

The enhancement filters are:

1. Laplacian- type filters.
2. Difference filter.

These filters will tend to bring out, or enhance details in the image. Example of convolution masks for the *Laplacian-type* filters are:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & -2 & 0 \\ -2 & 5 & -2 \\ 0 & -2 & 0 \end{bmatrix}$$

The ***Laplacian- type filters*** will enhance details in **all directions** equally**.** ***The difference filters*** will enhance details in the **direction specific to the mask selected**. There are four different filter convolution masks, corresponding to lines in the vertical, horizontal and two diagonal directions:

$$\begin{array}{cccc} \textbf{VERTICAL} & \textbf{HORIZANTAL} & \textbf{DIAGONAL1} & \textbf{DIAGONAL2} \\ \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \end{array}$$

Example: Apply the following on the following sub image:

$$\begin{bmatrix} 100 & 30 & 20 & 40 & 15 \\ 15 & 130 & 220 & 30 & 40 \\ 30 & 200 & 40 & 120 & 20 \\ 15 & 50 & 20 & 60 & 50 \end{bmatrix}$$

1- Minimum filter to remove salt-type noise.

Solution:

g(1,1)= $\underline{15}$ 20 30 30 40 100 130 200 220.
g(1,2)= $\underline{20}$ 30 30 40 40 120 130 200 220
g(1,3)= $\underline{15}$ 20 20 30 40 40 40 120 220
g(2,1)= $\underline{15}$ 15 20 30 40 50 130 200 220
g(2,2)= $\underline{20}$ 30 40 50 60 120 130 200 220
g(2,3)= $\underline{20}$ 20 30 40 40 50 60 120 220

The resultant sub image is:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 15 & 20 & 15 & 0 \\ 0 & 15 & 20 & 20 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2- Maximum filter to remove pepper-type noise.

Solution:

g(1,1)= 15 20 30 30 40 100 130 200 $\underline{220}$.
g(1,2)= 20 30 30 40 40 120 130 200 $\underline{220}$
g(1,3)= 15 20 20 30 40 40 40 120 $\underline{220}$
g(2,1)= 15 15 20 30 40 50 130 200 $\underline{220}$
g(2,2)= $\underline{20}$ 30 40 50 60 120 130 200 $\underline{220}$
g(2,3)= $\underline{20}$ 20 30 40 40 50 60 120 $\underline{220}$

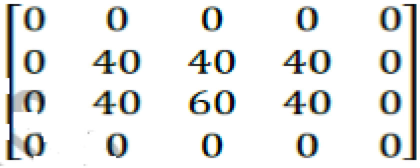
The resultant sub image is

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 220 & 220 & 220 & 0 \\ 0 & 220 & 220 & 220 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

3- Median filter to remove salt and pepper-type noise.
Solution:

g(1,1)= 15 20 30 30 $\underline{40}$ 100 130 200 220.
g(1,2)= 20 30 30 40 $\underline{40}$ 120 130 200 220
g(1,3)= 15 20 20 30 $\underline{40}$ 40 40 120 220
g(2,1)= 15 15 20 30 $\underline{40}$ 50 130 200 220
g(2,2)= 20 30 40 50 $\underline{60}$ 120 130 200 220
g(2,3)= 20 20 30 40 $\underline{40}$ 50 60 120 220

The resultant sub image is

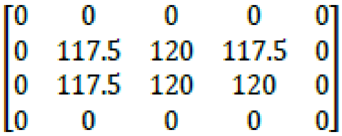


$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 40 & 40 & 40 & 0 \\ 0 & 40 & 60 & 40 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

3- Midpoint filter to remove Gaussian or Uniform noise.
Solution:

g(1,1)= $\underline{15}$ 20 30 30 40 100 130 200 $\underline{220}$.→ (15+220)/2=117.5
g(1,2)= $\underline{20}$ 30 30 40 40 120 130 200 $\underline{220.}$→(20+220)/2=120
g(1,3)= $\underline{15}$ 20 20 30 40 40 40 120 $\underline{220.}$ →(15+220)/2=117.5
g(2,1)= $\underline{15}$ 15 20 30 40 50 130 200 $\underline{220.}$ →(15+220)/2=117.5
g(2,2)= $\underline{20}$ 30 40 50 60 120 130 200 $\underline{220.}$→(20+220)/2=120
g(2,3)= $\underline{20}$ 20 30 40 40 50 60 120 $\underline{220.}$ →(20+220)/2=120

The resultant sub image is



$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 117.5 & 120 & 117.5 & 0 \\ 0 & 117.5 & 120 & 120 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

4- Mean filter

g(1,1)= 1/9(100+30+20+15+130+220+30+200+40)=87.2 ≅ 87
g(1,2)= 1/9(30+20+40+130+220+30+200+40+120)=92.2 ≅ 92
g(1,3)= 1/9(20+40+15+220+30+40+40+120+20)=60.5≅ 61
g(2,1)= 1/9(15+130+220+30+200+40+15+50+20)=80
g(2,2)= 1/9(130+220+30+200+40+120+50+20+60)=96.6≅ 97
g(2,3)= 1/9(220+30+40+40+120+20+20+60+50)=66.6≅ 67

The resultant sub image is

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 87 & 92 & 61 & 0 \\ 0 & 80 & 97 & 67 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

5- Laplacian filter with the following mask:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Solution:

g(1,1)= (100*0)+(30*-1)+(20*0)+(15*-1)+(130*5)+(220*-1)+(30*0)+(200*-1)+(40*0)=185

g(1,2)= (30*0)+(20*-1)+(40*0)+(130*-1)+(220*5)+(30*-1)+(200*0)+(40*-1)+(120*0)=880

=255

g(1,3)= (20*0)+(40*-1)+(15*0)+(220*-1)+(30*5)+(40*-1)+(40*0)+(120*-1)+(20*0)=-270 =0

g(2,1)= (15*0)+(130*-1)+(220*0)+(30*-1)+(200*5)+(40*-1)+(15*0)+(50*-1)+(20*0)=750=255

g(2,2)= (130*0)+(220*-1)+(30*0)+(200*-1)+(40*5)+(120*-1)+(50*0)+(20*-1)+(60*0)=-360=0

g(2,3)= (220*0)+(30*-1)+(40*0)+(40*-1)+(120*5)+(20*-1)+(20*0)+(60*-1)+(50*0)=450=255

The resultant sub image is

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 185 & 255 & 0 & 0 \\ 0 & 255 & 0 & 255 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

6- Difference filter that enhance the image in the vertical direction:

Solution:

The mask will use is

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

g(1,1)= (100*0)+(30*1)+(20*0)+(15*0)+(130*1)+(220*0)+(30*0)+(200*-1)+(40*0)=-40=0

g(1,2)= (30*0)+(20*-1)+(40*0)+(130*-1)+(220*5)+(30*-1)+(200*0)+(40*-1)+(120*0)=880

=255

g(1,3)= (20*0)+(40*-1)+(15*0)+(220*-1)+(30*5)+(40*-1)+(40*0)+(120*-1)+(20*0)=-200=0

g(2,1)= (15*0)+(130*-1)+(220*0)+(30*-1)+(200*5)+(40*-1)+(15*0)+(50*-1)+(20*0)=-380=0

g(2,2)= (130*0)+(220*-1)+(30*0)+(200*-1)+(40*5)+(120*-1)+(50*0)+(20*-1)+(60*0)=700 =0

g(2,3)=(220*0)+(30*-1)+(40*0)+(40*-1)+(120*5)+(20*-1)+(20*0)+(61*-1)+(50*0)=499=255

The resultant sub image is

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 255 & 0 & 0 \\ 0 & 0 & 0 & 255 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$