



Image Processing1

Third Class/ Multimedia Branch

By: Prof. Dr. Matheel Emaduldeen Abdulmunim

2023-2024

LECTURE 1

Introduction to Computer Vision and Image Processing (CVIP)

1. Computer Imaging

Can be defined as an 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 visual 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).

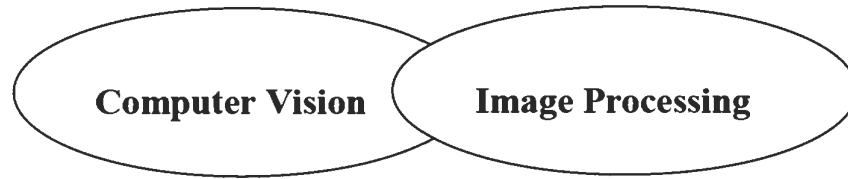


Figure (1): Computer imaging can be separated into two different but overlapping areas.

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

2. Computer Vision

Computer vision is computer imaging where the application does not involve a human being in visual loop. 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
2. Medical Community
3. Law Enforcement
4. Infrared Imaging
5. Satellites Orbiting.

3. Image Processing

Image processing is computer imaging where application involves a human being in the visual loop. In other words the image are to be examined and a acted upon by people.

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

1. Medical community
2. Computer – Aided Design
3. Virtual Reality
4. Image Processing.

The major topics within the field of image processing include:

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

4. 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.

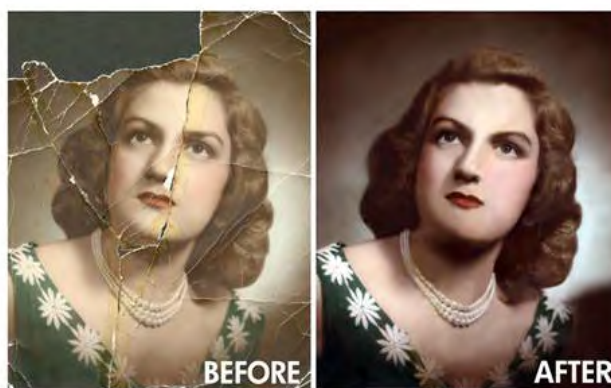


Figure (2): Image Restorationa. Image with distortionb. Restored image

5. 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 be suitable for enhancing medical images.

Although enhancement and restoration are similar in aim, to make an image look better. They differ in how they approach the problem. Restoration methods 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.

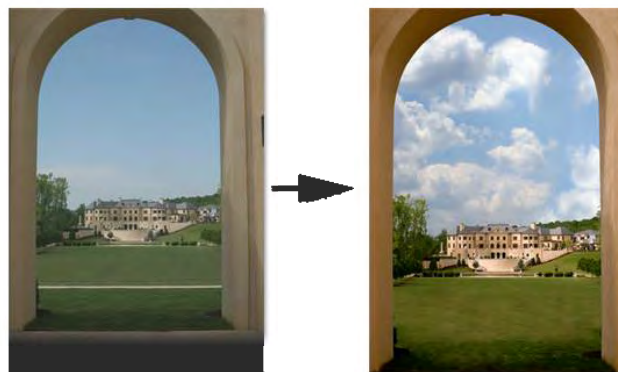


Figure (3): Image Enhancement

6. Image Compression

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



a. Image before compression 135 mm F13



b. Image after compression 200 mm F13

Figure(4): Image Compression.

LECTURE 2

7. Computer Imaging Systems

Computer imaging systems are comprised of two primary components types, hardware and software. The hardware 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.

8. 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 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.

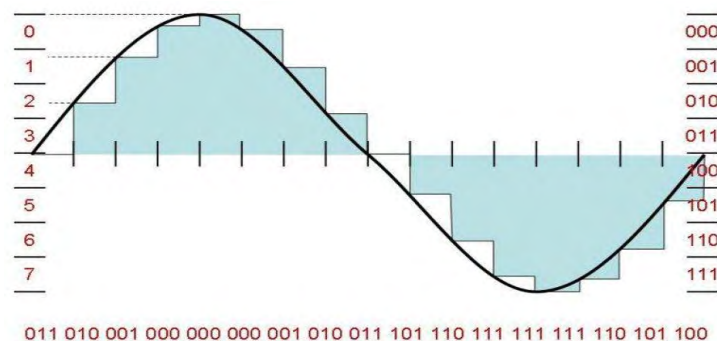


Figure (5): Digitizing (Sampling) an Analog Video Signal[1].

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.

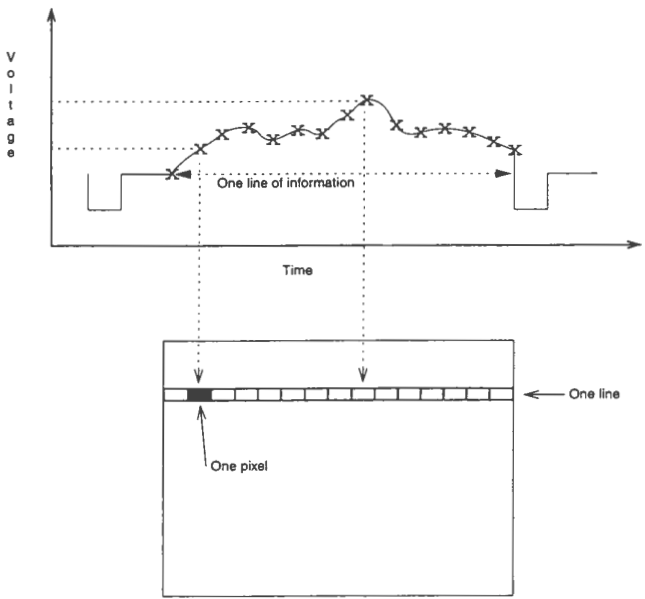


Figure (6): Digitizing (Sampling) an Analog Video Signal[2].

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 :

$$I(r, c) = \text{The brightness of image at the point } (r,c)$$

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:

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

In above image matrix, the image size is (NXN) [matrix dimension]

then: $N_g = 2^m \dots\dots\dots(1)$

Where N_g denotes the number of gray levels m is the no. of bits contains in digital image matrix.

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

Solution

$$N_{\text{gray}} = 2^6 = 64 \text{ Gray Level}$$

$$N_{\text{bit}} = 128 * 128 * 6 = 98304 \text{ bit} = 12\text{KB}$$

9. The Human Visual System

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

- 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 path ways for visual information to travel from the receiving sensor (the eye) to the processor (the brain).

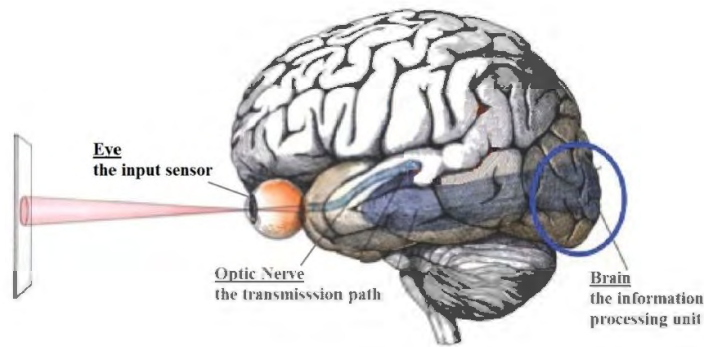


Figure (7): The Human Visual System.

10. Spatial Frequency Resolution

The image 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.

Spatial frequency concept, where frequency 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.

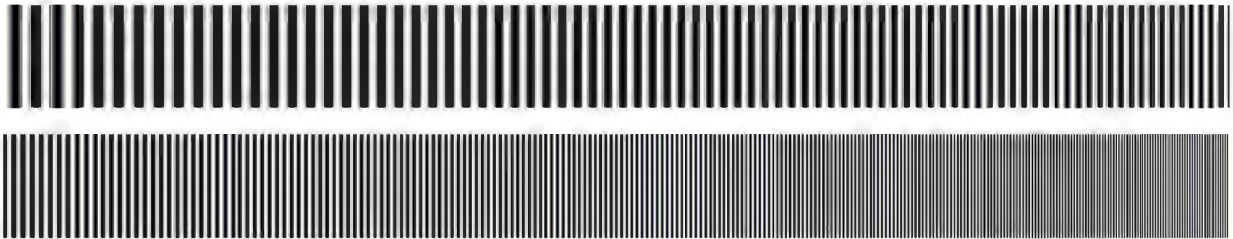


Figure (8) Resolution and Spatial Frequency.

11. Brightness Adaptation

In image we observe many brightness levels and the vision system can adapt to a wide range. If the mean value of the pixels inside the image is around Zero gray level then the brightness is low and the images dark but for mean value near the 255 then the image is light.



Figure (9): Brightness Adaptation.

If fewer gray levels are used, we observe false contours bogus lines resulting from gradually changing light intensity not being accurately represented.



Figure (10): Brightness Adaptation.

12. Image Representation

We have seen that the human visual system (HVS) receives an input image as a collection of spatially distributed light energy; this form is called an optical image. Optical images are the type we deal with everyday 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-Binary Images.

2-Gray Scale Images.

3-Color Images.

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 check a manufactured object for deformation, in optical character recognition (OCR).

Binary images are often created from gray-scale images via a threshold value is turned white (1), and those below it are turned black (0).



Figure (11): Binary Images.

2. Gray Scale Image

Gray_scale images are referred to as monochrome, or one-color image. They contain brightness information only brightness information only, no color information. The number of different brightness level available. 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.

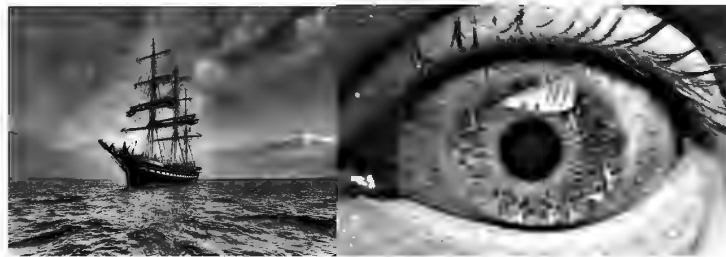


Figure (12): Gray Scale Images.

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.



Figure (13): Color Images.

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.

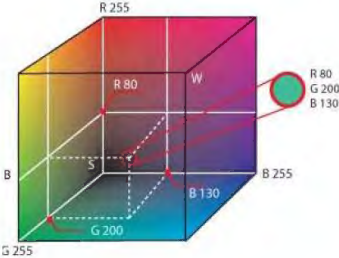


Figure (14) Typical RGB color image can be thought as three separate images $IR(r,c),IG(r,c),IB(r,c)$.

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 $-(R,G,B)$.

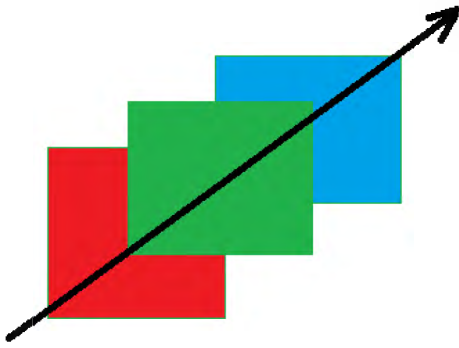


Figure (15): A color pixel vector consists of the red, green and blue pixel values (R, G, B) at one given row/column pixel coordinate(r,c).

For many applications, RGB color information is transformed into mathematical space that decouples the brightness information from the color information. The hue/saturation/lightness (HSL) color transform allows us to describe colors in terms that we can more readily understand.

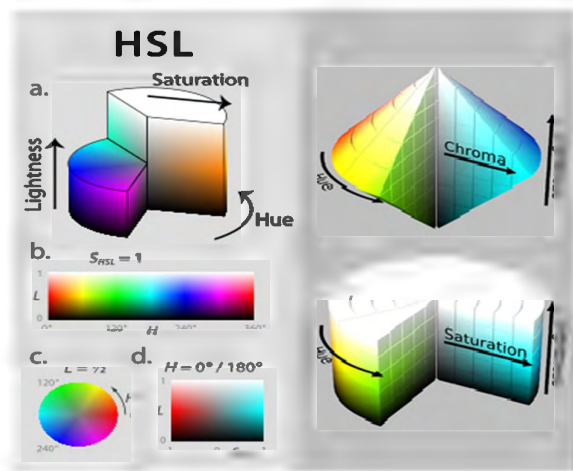


Figure (16): HSL Color Space.

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.Modeling the color information creates a more people oriented way of describing the colors.

4. Multispectral Images

Multispectral images typically contain information outside the normal human perceptual range. This may include infrared, ultraviolet, X-ray, acoustic or radar data. Source of these types of image include satellite systems underwater sonar systems and medical diagnostics imaging systems.



Figure (17): Multispectral images.

13. 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.

Computer Graphics :

Computer graphics is a specialized field within that refers to the computer science realm that refers to the reproduction of visual data through the use of 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 bit map format where each pixel has specific values associated with it. Most the type of file format fall into category of bitmap images. In general, these types of images contain both header information and the raw pixel data.

The header information contain 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, $I(r,c)$.

Image File Format :

1. BMP format:

It is the format used by the windows, 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).

2. Bin file format:

It is the raw image data $I(r,c)$ with no header information.

3. PPM file format:

It contain raw image data with simplest header, the PPM format, include PBM(binary),PGM(gray),PPM(color), PNM(handles any of the previous type) the header contain a magic number that identifies and determines information.

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

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.

5. JPEG (Joint photo Graphic Experts Group):

It is simply becoming standard that allows images compressed algorithms to be used in many different computer platforms. JPEG images compression is being used extensively on the WWW. It's, flexible, so it can create large files with excellent image equality.

6. 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 rang (0-225).

LECTURE 3

Image Analysis Preprocessing

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 questions: 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 end 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.

1. System Model :

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

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

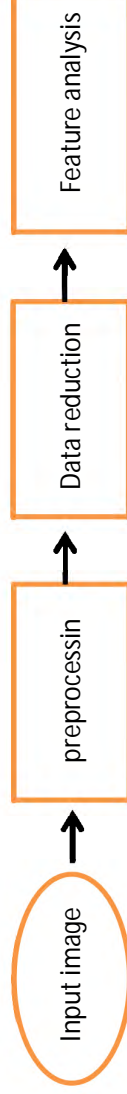


Figure (1): image analysis .

1. Preprocessing :

Is used to remove noise and eliminate irrelevant, visually unnecessary information. Noise is unwanted information that can result from the image acquisition process, other preprocessing steps might include:

- Gray –level or spatial quantization (reducing the number of bits per pixel or the image size).
- Finding regions of interest for further processing.

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 examined 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.

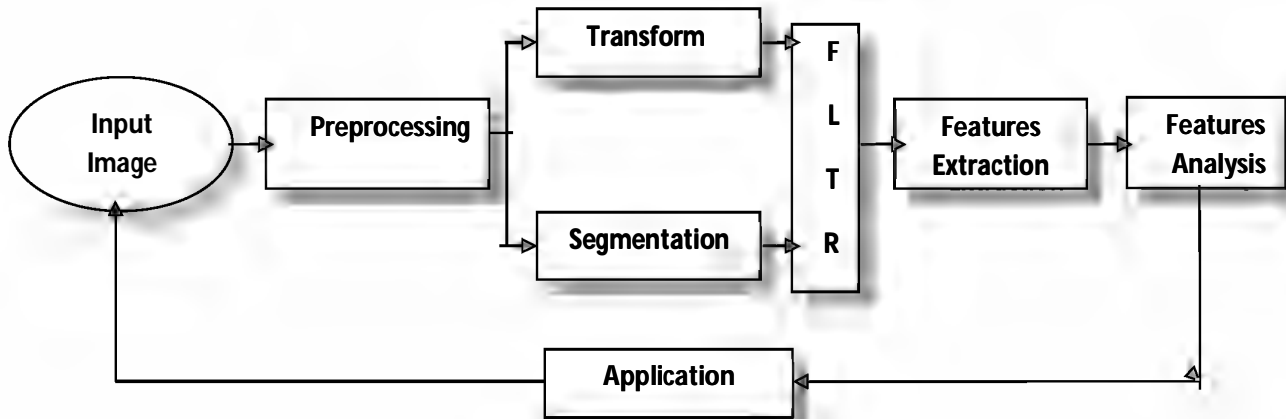


Figure (2): Image Analysis

2. Preprocessing

The preprocessing algorithm, techniques and operators are used to perform initial processing that makes the primary data reduction and analysis task easier. They include operations related to:

- Extracting regions of interest.
- Performing basic algebraic operation on image.
- Enhancing specific image features.
- Reducing data in resolution and brightness.

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.

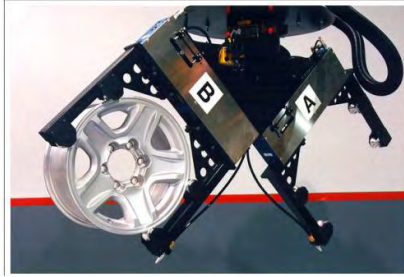


Figure (3): a robot gripper.

3. Region of Interest Image Geometry :

Often, for image analysis we want to investigate more closely a specific 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,Rotate.

The image crop process 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 enlarge it. The zoom process can be done in numerous ways:

1. **ZeroOrder Hold:** is performed by repeating previous pixel values, thus creating a blocky effect as in the following figure (4) :

we can do the zero order hold as follows:

Original Image Array

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

(NXN)(3 X 3)

Image with Rows Expanded

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

Image with rows and columns expanded

| | | | | | |
|---|---|---|---|---|---|
| 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 |

(2N x 2N) (6x6)

In zero order hold the output image size is double original image size (2n x2n) , which (n x n) is the dimension of image .

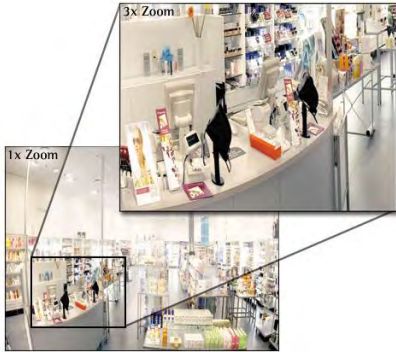


Figure (4) Zero Order Hold Method .

2. **First Order Hold:** is performed by finding linear interpolation between adjacent pixels, 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.

Next, take result and expanded the columns in the same way as follows:

Image with rows and columns expanded

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

$$(2N-1) \times (2N-1) (5 \times 5)$$

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

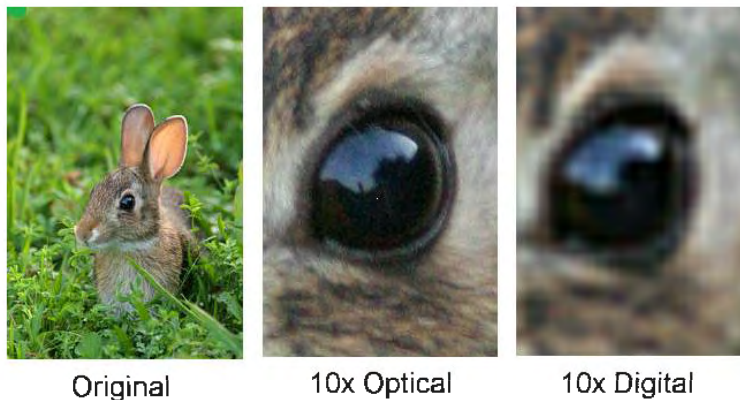


Figure (5): First _Order Hold.

3- 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.

The image is extended as follows:

Original Image Array

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

Image extended with zeros

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

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

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

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 one 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.

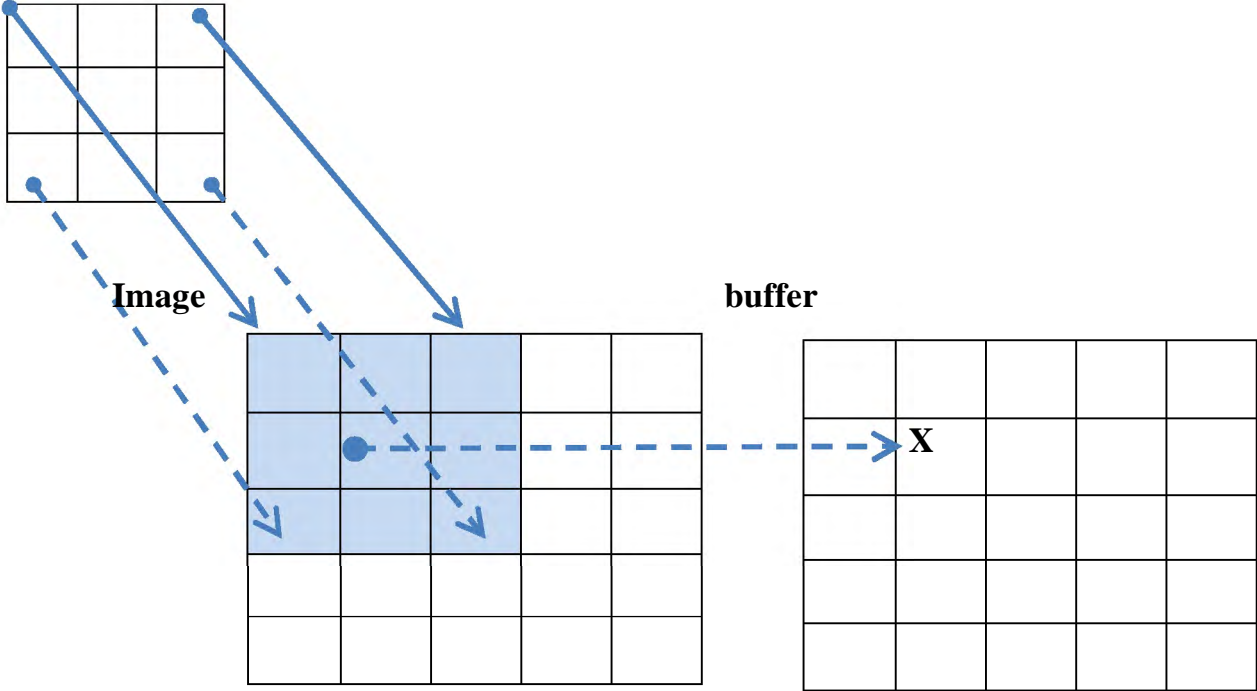


Figure (6) First Order Hold Method .

Note that 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.

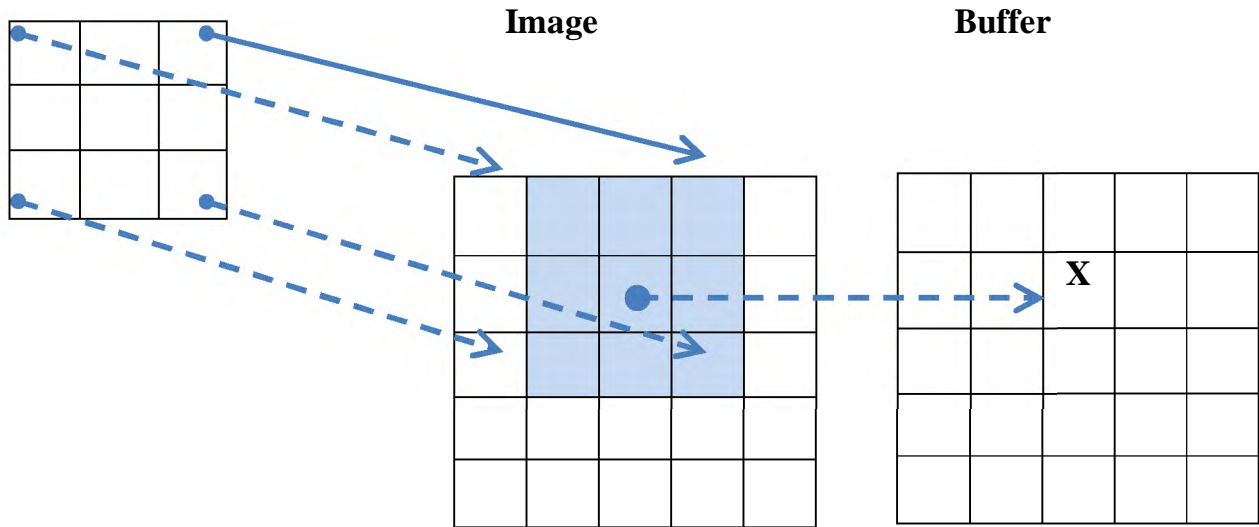
❖ **The convolution process :**

Mask



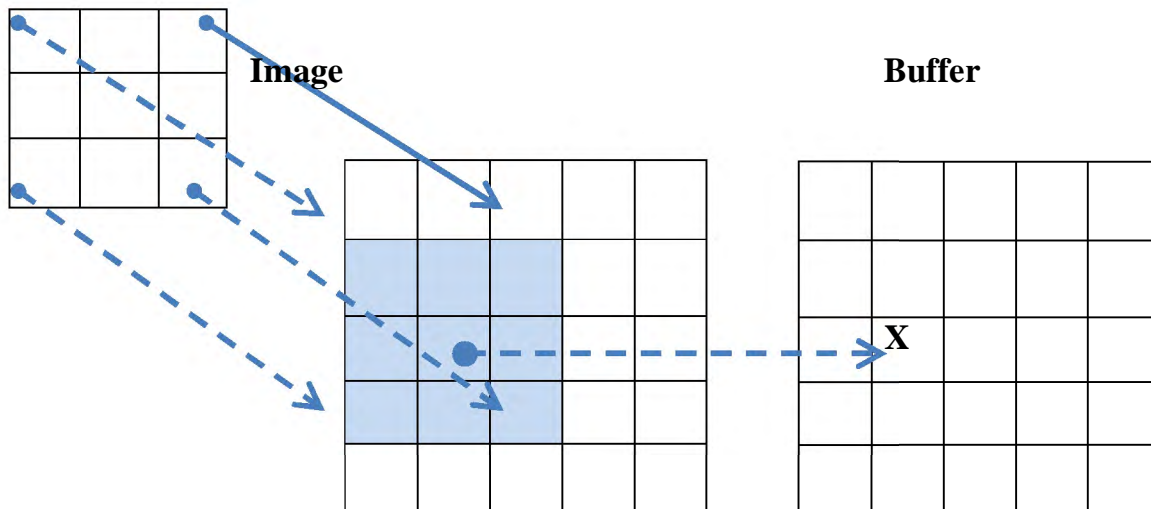
- a. Overlay the convolution mask in the upper-left corner of the image. Multiply coincident terms, sum, and put the result into the image buffer at the location that corresponds to the masks current center, which is $(r,c)=(1,1)$.

Mask



b. 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.

Mask



c. 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 column(s).

Example: Enlarge the following image using convolution mask for first order hold.

$$\begin{bmatrix} 3 & 5 \\ 2 & 7 \end{bmatrix}$$

Solution :-

a . Extend the image by adding rows and columns of zeros between the existing rows and columns and the image is extended as follows:

Image extended with zeros

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

b . Perform the convolution :

$$1) \left[\frac{1}{4} * 0 + \frac{1}{2} * 0 + \frac{1}{4} * 0 + \frac{1}{2} * 0 + 1 * 3 + \frac{1}{2} * 0 + \frac{1}{4} * 0 + \frac{1}{2} * 0 + \frac{1}{4} * 0 \right] = 3$$

$$2) \left[\frac{1}{4} * 0 + \frac{1}{2} * 0 + \frac{1}{4} * 0 + \frac{1}{2} * 3 + 1 * 0 + \frac{1}{2} * 5 + \frac{1}{4} * 0 + \frac{1}{2} * 0 + \frac{1}{4} * 0 \right] = 1.5 + 2.5 = 4$$

$$3) \left[\frac{1}{4} * 0 + \frac{1}{2} * 0 + \frac{1}{4} * 0 + \frac{1}{2} * 0 + 1 * 5 + \frac{1}{2} * 0 + \frac{1}{4} * 0 + \frac{1}{2} * 0 + \frac{1}{4} * 0 \right] = 5$$

$$4) [1/4 *0 +1/2 *3 +1/4 *0 +1/2 *0 +1*0 +1/2 *0 +1/4 *0 +1/2 *2 +1/4 *0] = 1.5 + 1 = 2.5$$

$$5) [1/4 *3 +1/2 *0 +1/4 *5 +1/2 *0 +1*0 +1/2 *0 +1/4 *2 +1/2 *0 +1/4 *7] = 0.75 + 1.25 +0.5 +1.75 = 4.25$$

$$6) [1/4 *0 +1/2 *5 +1/4 *0 +1/2 *0 +1*0 +1/2 *0 +1/4 *0 +1/2 *7 +1/4 *0] = 2.5 + 3.5 = 6$$

$$7) [1/4 *0 +1/2 *0 +1/4 *0 +1/2 *0 +1*2 +1/2 *0 +1/4 *0 +1/2 *0 +1/4 *0] = 2$$

$$8) [1/4 *0 +1/2 *0 +1/4 *0 +1/2 *2 +1*0 +1/2 *7 +1/4 *0 +1/2 *0 +1/4 *0] = 1 + 3.5 = 4.5$$

$$9) [1/4 *0 +1/2 *0 +1/4 *0 +1/2 *0 +1*7 +1/2 *0 +1/4 *0 +1/2 *0 +1/4 *0] = 7$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 4 & 5 & 0 \\ 0 & 2.5 & 4.25 & 6 & 0 \\ 0 & 2 & 4.5 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Enlarge image by convolution mask for first order hold

Why we use this convolution method when it require, so many more calculation than the basic averaging of the neighbors method?

The answer is that many computer boards can perform convolution in hardware, which is generally very fast, typically much faster than applying a faster algorithm in software. Not only first order hold be performed via convolution, but zero order hold

can also be achieved by extending the image with zeros and using the following convolution mask.

Zero-order hold convolution mask

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

Note that for this mask we will need to put the result in the pixel location corresponding to the lower-right corner because there is no center pixel.

Example: Enlarge the following image using convolution mask for zero order hold.

a . Extend the image by adding rows and columns of zeros between the existing rows and columns and the image is extended as follows :

Image extended with zeros

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

b . Perform the convolution :

- 1) $1*0 + 1*0 + 1*0 + 1*3 = 3$
- 2) $1*0 + 1*0 + 1*3 + 1*0 = 3$
- 3) $1*0 + 1*0 + 1*0 + 1*5 = 5$
- 4) $1*0 + 1*0 + 1*5 + 1*0 = 5$
- 5) $1*0 + 1*3 + 1*0 + 1*0 = 3$
- 6) $1*3 + 1*0 + 1*0 + 1*0 = 3$

$$7) 1*0 + 1*5 + 1*0 + 1*0 = 5$$

$$8) 1*5 + 1*0 + 1*0 + 1*0 = 5$$

$$9) 1*0 + 1*0 + 1*0 + 1*2 = 2$$

$$10) 1*0 + 1*0 + 1*2 + 1*0 = 2$$

$$11) 1*0 + 1*0 + 1*0 + 1*7 = 7$$

$$12) 1*0 + 1*0 + 1*7 + 1*0 = 7$$

$$13) 1*0 + 1*2 + 1*0 + 1*0 = 2$$

$$14) 1*2 + 1*0 + 1*0 + 1*0 = 2$$

$$15) 1*0 + 1*7 + 1*0 + 1*0 = 7$$

$$16) 1*7 + 1*0 + 1*0 + 1*0 = 7$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 3 & 5 & 5 \\ 0 & 3 & 3 & 5 & 5 \\ 0 & 2 & 2 & 7 & 7 \\ 0 & 2 & 2 & 7 & 7 \end{bmatrix}$$

Enlarge image by convolution mask for zero order hold

LECTURE 4

ECT

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 defining 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: we want to enlarge an image to five times its original size, and we have two adjacent pixel values 210 and 240.

1. Find the difference between the two values, $240 - 210 = 30$.
2. The desired enlargement is $k=5$, so we get $30/5 = 6$.
3. Next determine how many intermediate pixel values we need:
 $K-1 = (5-1) = 4$. The four pixel values between the 210 and 240 are:

Where :

$K = 1$:

$$210 + (6*1) = 216 .$$

$K = 2$:

$$210 + (6*2) = 222 .$$

$K = 3$:

$$210 + (6*3) = 228 .$$

$K = 4$:

$$210 + (6*4) = 234 .$$

- We do this for every pair of adjacent pixels first along the rows and then along the columns. This will allow us to enlarge the image by any factor of $K(N-1) + 1$ where K is an integer and $N \times N$ is the image size.
- To process opposite to enlarging an image is shrinking. This process is done by reducing the amount of data that need to be processed.
- Two other operations of interest image geometry are: Translation and Rotation. These processes may be performed for many application specific reasons, for example to align an image with a known template in pattern matching process or make certain image details easier to see.

The translation process can be done in the following equations :

$$R' = R + R_0$$

$$C' = C + C_0$$

Where R' and C' are new coordinates , R and C are the original coordinates and R_0 and C_0 are the distances to move or translate the image .

The rotation process requires the use of these equations :

$$R^{\wedge} = R (\cos\theta) + C (\sin\theta)$$

$$C^{\wedge} = -R (\sin\theta) + C(\cos\theta)$$

Where R^{\wedge} and C^{\wedge} are the new coordinates , R and C are the original coordinates , θ is the angle to rotate the image , θ is defined in a clockwise direction from the horizontal axis at the image origin in the upper left corner .

The rotation and translation process can be combined into one set of equations :

$$R^{\wedge} = (R + R_0) (\cos\theta) + (C + C_0) (\sin\theta)$$

$$C^{\wedge} = - (R + R_0) (\sin\theta) + (C + C_0) (\cos\theta)$$

Where R^{\wedge} and C^{\wedge} are the new coordinates and R, C, R_0, C_0 , and θ are previously defined .

There are some practical difficulties with direct application of these equations when translating , if we move everything one row down , what do we put in the top row ? there are two basic options .

- 1- Fill the top row with a constant value typically black (0) or white (255) .

2- Warp around by shifting the bottom row to the top , shown in figure (6)

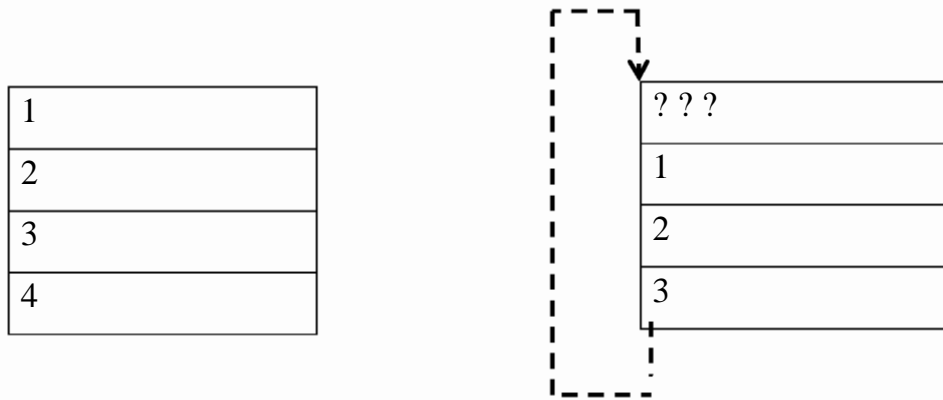


Figure (6) Translation .

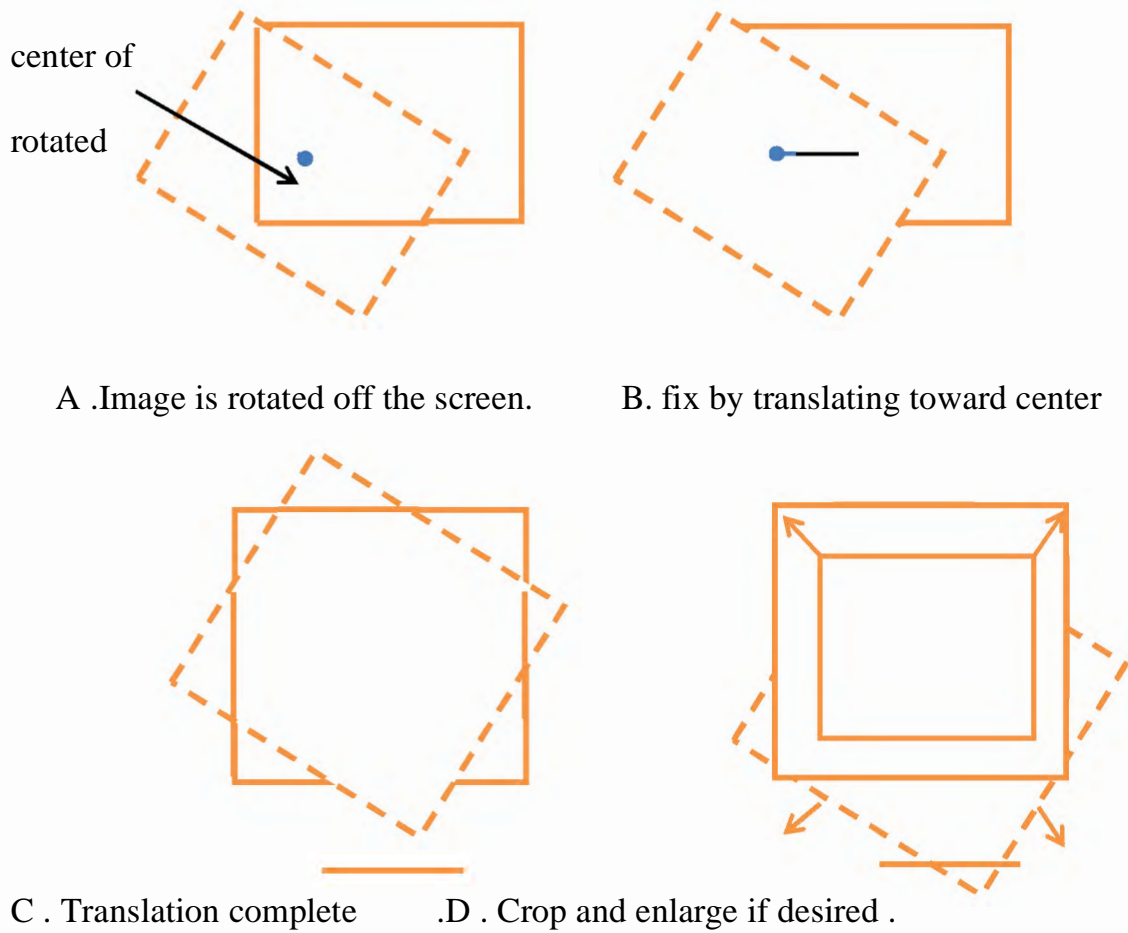


Figure (7) Rotation .

4. Image Algebra :

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

1. Arithmetic operations.
2. Logic operations.

Addition, subtraction, division and multiplications comprise the arithmetic operations, while AND, OR and NOT makeup the logic operations. These operations which require only one image, and are done on a pixel by pixel basis.

To apply the arithmetic operations to two images, we simply operate on corresponding pixel values. For example to add image i1 and i2 to create i3 :

$$\begin{array}{ccc} i1 & & i2 & & i3 \\ \left[\begin{array}{ccc} 3 & 4 & 7 \\ 3 & 4 & 5 \\ 2 & 4 & 6 \end{array} \right] & + & \left[\begin{array}{ccc} 6 & 6 & 6 \\ 4 & 2 & 6 \\ 3 & 5 & 5 \end{array} \right] & = & \left[\begin{array}{ccc} 3+6 & 4+6 & 7+6 \\ 3+4 & 4+2 & 5+6 \\ 2+3 & 4+5 & 6+5 \end{array} \right] & = & \left[\begin{array}{ccc} 9 & 10 & 13 \\ 7 & 6 & 11 \\ 5 & 9 & 11 \end{array} \right] \end{array}$$

- Addition is used to combine the information in two images. Applications include development of image restoration algorithm for molding additive noise, and special effects, such as image morphing in motion pictures.
- Subtraction 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. Applications include Object tracking, Medical imaging, Law enforcement and Military applications .
- Multiplication and Division are used to adjust the brightness of an image. One image typically consists of a constant number greater than one. Multiplication of

the pixel values by a number greater than one will darken the image (Brightness adjustment is often used as a processing step in image enhancement).

The logic operations AND, OR and NOT form a complete set, meaning that any other logic operation (XOR, NOR, NAND) can be created by a combination of these basic elements. They operate in a bit-wise fashion on pixel data.

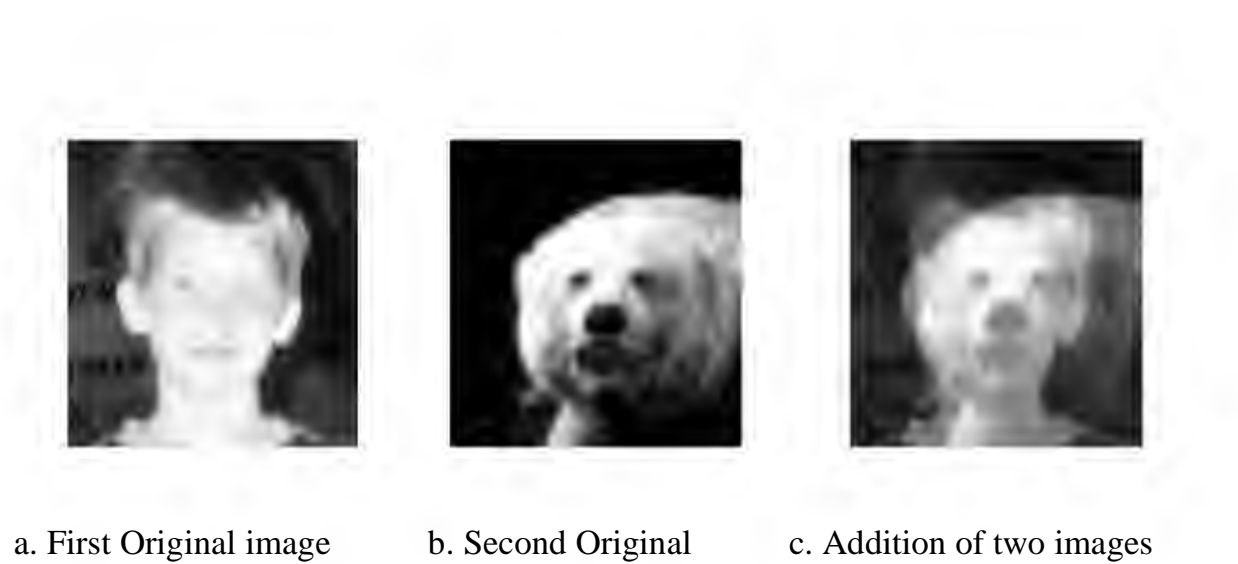


Figure (7) Image Addition.

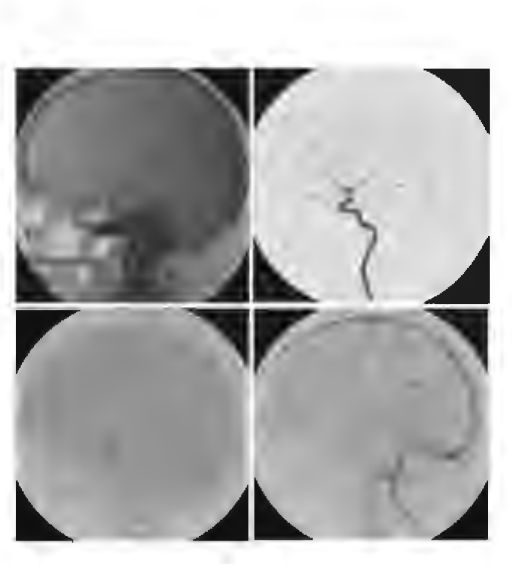


Figure (8) Image Subtraction .



Figure (9) Image Multiplication.



Figure (10) Image Division.

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

$(111)_{10} \longrightarrow 011011112$

AND

$(88)_{10} \longrightarrow 01011000 2$

01001000

The logic operation AND and OR are used to combine the information in two images. They may be done for special effects, but a more useful application for image analysis

is to perform a masking operation. Use AND and OR as a simple method to extract a Region of Interest from an image, if more sophisticated graphical methods are not available.

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. The NOT operation creates a negative of the original image, by inverting each bit within each pixel value (every '0' convert to '1' and every '1' convert to '0').



a. Original image



b. Image mask (AND)



c. ANDing a and b



d. Image mask (OR)



e. ORing a and d

Figure (11) Image masking.

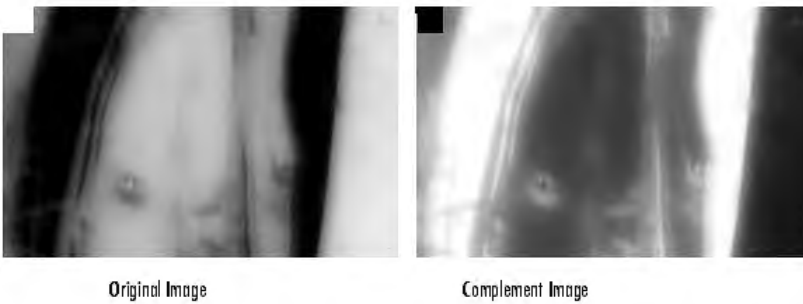


Figure (12) Complement Image.

5. Image Restoration:

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.

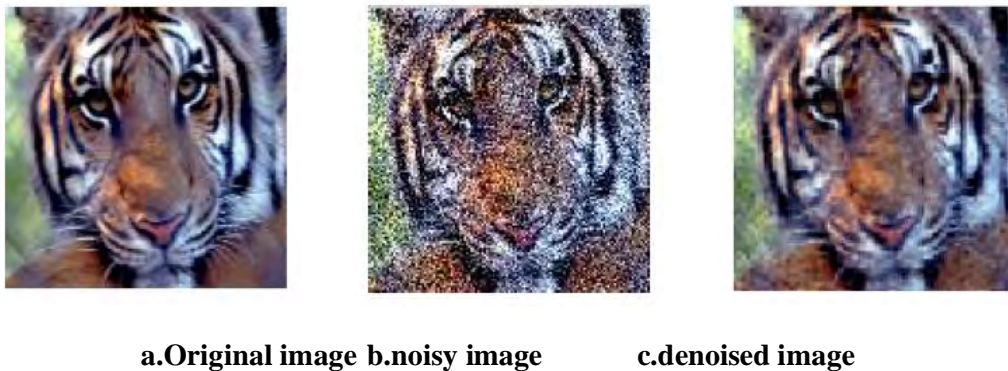


Figure (13) :Image denoising.

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.

The three types of filters are:

1. Mean filters .
2. Median filters (order filter) .
3. Enhancement filters .

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 filter high lights edges and details within the image.

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.

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

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

Not 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 smoothies 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.



a. Original image



b. Mean filtered image

Figure (14) Mean Filter.

The median filter is a nonlinear 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 center 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.

$$I_1 \leq I_2 \leq I_3 \dots \dots \dots < I_N$$

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



a. Salt and pepper noise



b. Median filtered image (3x3)

Figure (15) Median Filter.

Example: Given the following 3X3 neighborhood :

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

We first sort the value in order of size (3,3,4,4,5,5,5,6,7) ; then we select the middle value , in this case it is 5. This 5 is then placed in center location.

A median filter can use a neighborhood of any size, but 3X3, 5X5 and 7X7 are typical. Note that the output image must be written to a separate image (buffer), so that the results are not corrupted as this process is performed.

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

The enhancement filters are:

1. Laplacian type.
2. Difference filter.

These filters will tend to bring out, or enhance details in the image.

Two 3x3 convolution masks for the laplacian type filters are :

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{pmatrix}$$

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.

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

Vertical **Horizontal** **Diagonal 1** **Diagonal 2**

LECTURE 5

ECT

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 themselves $I(r, c)$).
2. Spatial reduction (reduce the spatial coordinate (r, c)).

The simplest method of gray level reduction is **Thresholding**. We select a threshold gray level and set everything above that value equal to “1” and everything below the threshold equal to “0”. This effectively turns a gray level image into a binary (two level) image and is often used as a preprocessing step in the extraction of object features, such as shape, area, or perimeter.

A more versatile method of 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 1: We want to reduce 8bit information containing 256 possible gray level values down to 32 possible values.

$$8 \text{ bit} = 2^3$$

This can be done by ANDing each 8-bit value with the bit string **11111000**.

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:

$$256 \div 8 = 32$$

The general case requires us to mask k bits, where (2^k) is divided into the original graylevel 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 level, this can be done by ANDing each 8-bit value with bit string 11111110 (2^1).

❖ Image quantization by masking to 64 graylevel. 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.

Example 2 : We want to reduce 8-bit information containing (256 possible gray level) value down to 4 possible values.

AND-MASK

Sol.\

1- Determine (n) value:

256 gray level \longrightarrow 4 gray levels.

$$2^8 \longrightarrow 2^2$$

$$n = 2$$

2- Extract mask:

$$\begin{aligned} \text{mask} &= 256 - 2^{8-n} \\ &= 256 - 2^{8-2} \\ &= 256 - 2^6 = 256 - 64 \end{aligned}$$

mask = 192

Let g = 212

$$2^7 \ 2^6 \ 2^5 \ 2^4 \ 2^3 \ 2^2 \ 2^1 \ 2^0$$

| | | | | | | | | |
|-----|----|----|----|---|---|---|---|-------|
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | = 212 |

And

| | | | | | | | | |
|---|---|---|---|---|---|---|---|-------|
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | = 192 |
|---|---|---|---|---|---|---|---|-------|

| | | | | | | | | |
|---|---|---|---|---|---|---|---|-------|
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | = 192 |
|---|---|---|---|---|---|---|---|-------|

3-Shift to right

$$\begin{aligned} \text{no. of shift right} &= 8 - n \\ &= 8 - 2 = 6 \end{aligned}$$

>> 6

| | | | | | | | | |
|---|---|---|---|---|---|---|---|-----|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | = 3 |
|---|---|---|---|---|---|---|---|-----|

← One of black level

$$x = \text{result value} * \frac{256}{4}$$

$$= 3 * \frac{256}{4}$$

$$= 192 \quad \leftarrow \text{One of white level}$$

Example 3 : We want to reduce 8-bit information containing (256 possible gray level) value down to 4 possible values.

OR – MASK

Sol.\

1- Determine (n) value:

256 gray level \longrightarrow 4 gray level

$$2^8 \longrightarrow 2^2$$

$$n = 2$$

2- Extract mask:

$$\begin{aligned} \text{mask} &= 2^{8-n} - 1 \\ &= 2^{8-2} - 1 \\ &= 2^6 - 1 = 64 - 1 = 63 \end{aligned}$$

mask = 63

Let g = 212

| 2^7 | 2^6 | 2^5 | 2^4 | 2^3 | 2^2 | 2^1 | 2^0 | |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | = 212 |

OR

| | | | | | | | | |
|---|---|---|---|---|---|---|---|------|
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | = 63 |
|---|---|---|---|---|---|---|---|------|

| | | | | | | | | |
|---|---|---|---|---|---|---|---|-------|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | = 255 |
|---|---|---|---|---|---|---|---|-------|

3- Shift to right

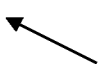
no. of shift right = $8 - n$

$$= 8 - 2 = 6$$

| | | | | | | | | |
|---|---|---|---|---|---|---|---|--|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
|---|---|---|---|---|---|---|---|--|

>> 6

| | | | | | | | | |
|---|---|---|---|---|---|---|---|-----|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | = 3 |
|---|---|---|---|---|---|---|---|-----|



One of black level

$$x = \text{result value} * \frac{256}{4}$$

$$= 3 * \frac{256}{4}$$

$$= 192$$

One of white level

Contouring appears in the image as false edges, or lines as a result of the gray level quantization method.



Original 8-bit image,
256 gray levels



Quantized to 6 bits,
64 gray levels



Quantized to 3 bits,
8 gray levels



Quantized to 1 bits,
2 gray levels

Figure (16) False Contouring .

This 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 (17) IGS quantization .

LECTURE 6

EDGE LINE DETECTION

1. Define of edge / line detection

The edge and line detection operators presented here represent the various types of operators used today. They are implemented with convolution masks and most are based on discrete approximations to differential operators. Edge detection methods are used as a first step in the line detection process.

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

And something about their texture.

Also edge detection used to find complex object boundaries by marking potential edge point corresponding to place in an image where rapid change in brightness occur. The noise in image can be creating problems so to solve this problem well first make preprocess to image.

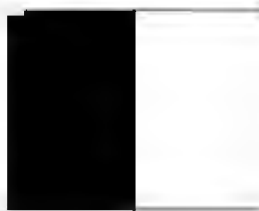
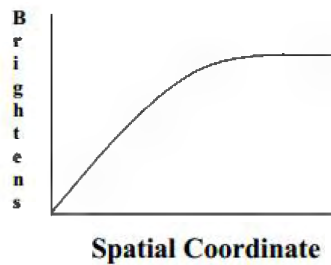
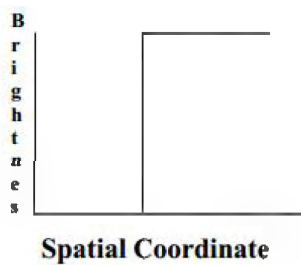


2. Methods of detection edge / line

Edge detection operators are based on the idea that edge information in an image is found by looking at the relationship of pixels with its neighbors. So if their gray-level value is similar with these around it “actual there is no edge “and their edge with neighbor are widely different in gray-level “that is mean actually it is edge found”. In practice, edges are caused by: Change in color or texture, Specific lighting conditions present during the image acquisition process.



The different between the real edge and ideal edge



a. Ideal Edge



b. Real Edge

The vertical axis represents brightness, and the horizontal axis shows the spatial coordinates. The abrupt change in brightness characterizes an ideal edge. In the

figure above 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.

There are two basic principles for each edge detector mask:

- **First:** the number in the mask sum to zero. If 3×3 areas of an image contains a constant value (such as all ones), then there are no edges in that area. The result of convolving that area with a mask should be zero. If the numbers in the mask sum to zero, then convolving the mask with a constant area will result in the correct answer of zeros.
- **Second:** the masks should approximate differentiation or amplify the slope of the edge. The simple example $[-1 \ 0 \ 1]$ given earlier showed how to amplify the slope of the edge.

The number of masks used for edge detection is almost limitless. Research have used different techniques to derive masks, some of will be illustrated in the following section.

An edged is where the gray level of the image moves from an area of low Values to high values or vice versa. The edge itself is at the centre of this transition. To detect the edge/line detection we need to use same mask to detect the edge or line in any image. There are many masks use to detect the edge/line .The masks are:

1. Roberts operator
2. Sobol operator
3. Prewitt operator
4. Kirsch compass masks

5. Robinson compass masks
6. Laplacian operator
7. Fre-chen masks
8. Edge operator performance
9. Hough transform

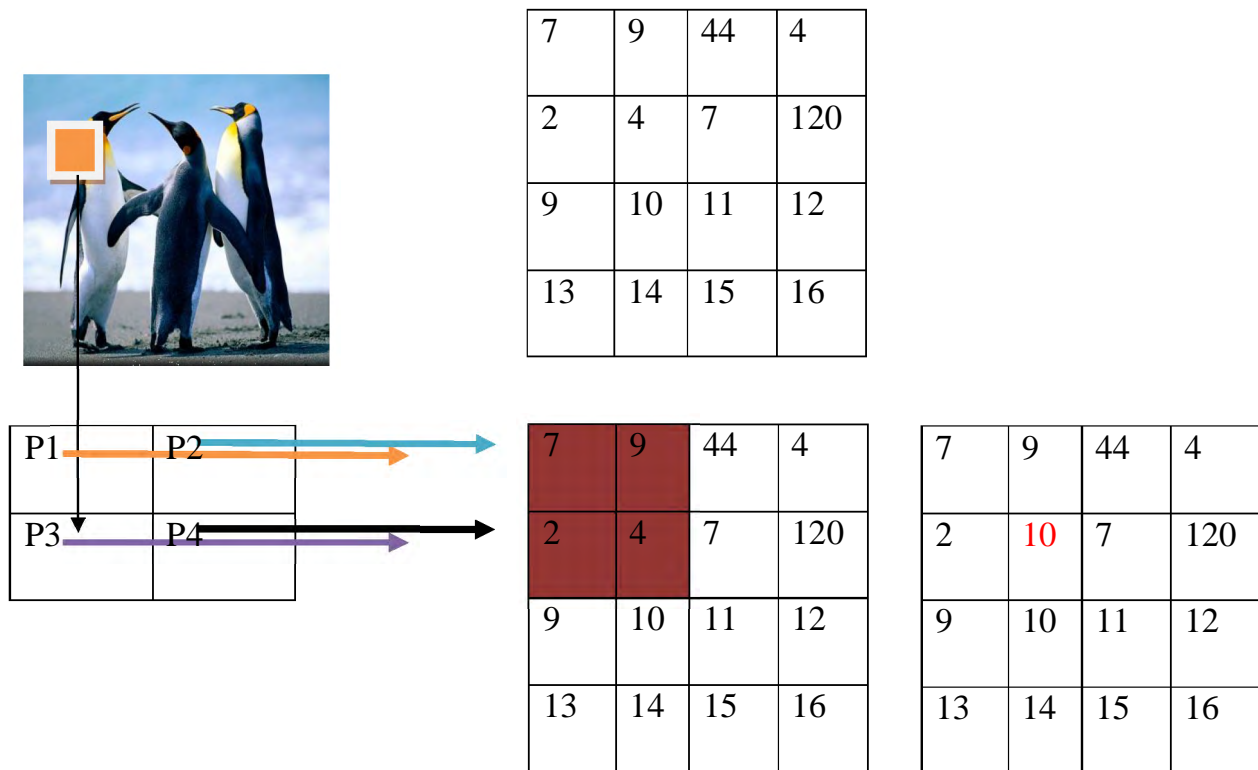
1-Roberts operator:

The Roberts operator mark edge only, it dose not return any information about the edge orientation. It is simplest for edge detection and it is work best with binary edge.

There are two form of operator. First form of reports operator is:

$$\text{New-pixel} = \left[|I(r,c) - I(r-1,c-1)| + |I(r,c-1) - I(r-1,c)| \right]$$

Example: suppose we have this image



New-pixel= [(p1 - p4) +(p2-p3)] then

New-pixel=7-4+9-2 = 10 Then shift one pixel to right

| | | | |
|----|----|----|-----|
| 7 | 9 | 44 | 4 |
| 2 | 4 | 7 | 120 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

New - pixel = [(9-7)+(44-4)]

= 42

| | | | |
|----|----|----|-----|
| 7 | 9 | 44 | 4 |
| 2 | 10 | 42 | 120 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

Then complete this formula to all images.



2- Sobel operator: The sobel edge detection look for edge in both direction (H and V) then combine this information into single metric. Sobel edge used two masks the:

The horizontal mask =
$$\begin{bmatrix} -1, -2, -1 \\ 0, 0, 0 \\ 1, 2, 1 \end{bmatrix}$$

The vertical =
$$\begin{bmatrix} -1, 0, 1 \\ -2, 0, 2 \\ -1, 0, 1 \end{bmatrix}$$

And then find the magnitude of edge by

$$\sqrt{(x * x) + (y * y)}$$

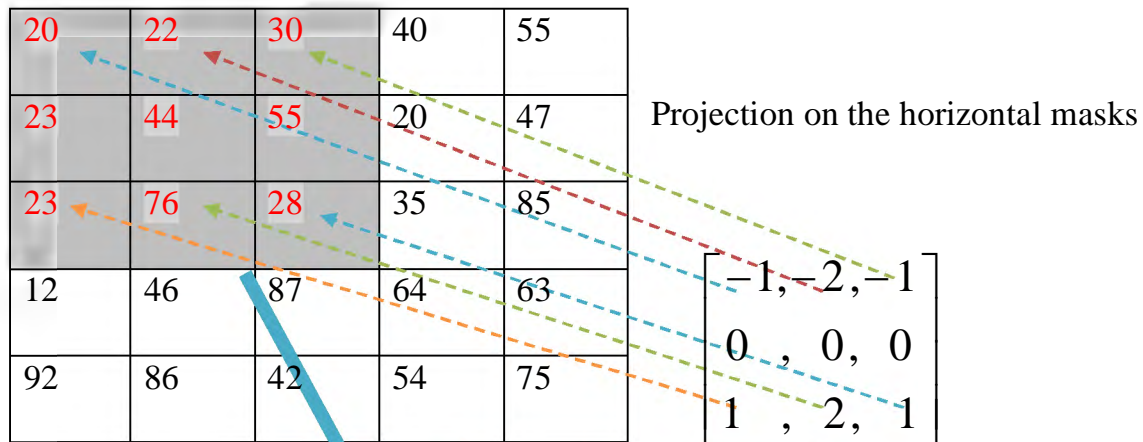
And find the direction of edge

$$\text{Tan}^{-1} \left[\frac{(x * x)}{(y * y)} \right]$$

Example: If we won't to compute the edge for this image

| | | | | |
|----|----|----|----|----|
| 20 | 22 | 30 | 40 | 55 |
| 23 | 44 | 55 | 20 | 47 |
| 23 | 76 | 28 | 35 | 85 |
| 12 | 46 | 87 | 64 | 63 |
| 92 | 86 | 42 | 54 | 75 |

To find the edge by sobel



New - pixel =

| | | |
|-----------|-----------|-----------|
| p1 | p2 | p3 |
| p4 | p5 | p6 |
| p7 | p8 | p9 |

$$\text{New-pixel} = p1+p2+p3+p7+p8+p9$$

$$-1*20+(-2*22)+(-1*30)+1*23+(2*76)+1*28=-20-44-30+23+152+28=109$$

| | | | | |
|----|----|----|----|----|
| 20 | 22 | 30 | 40 | 55 |
| 23 | 44 | 55 | 20 | 47 |
| 23 | 76 | 28 | 35 | 85 |
| 12 | 46 | 87 | 64 | 63 |
| 92 | 86 | 42 | 54 | 75 |

$$\begin{bmatrix} -1, 0, 1 \\ -2, 0, 2 \\ -1, 0, 1 \end{bmatrix}$$

Then find the vertical

$$\text{New-pixel} = -20 - (23 + 23) - 23 + 30 + (55 + 55) + 28 = 125$$

Then when we find the horizontal and vertical edge we can then find the edge magnitude and direction

109 for x and 125 for y

$$\text{Edge magnitude} = \sqrt{(x * x) + (y * y)}$$

$$= 109 * 109 + 125 * 125 = 165.8$$

$$\text{Edge direction} = \tan^{-1}(109 * 109 / 125 * 125) = \tan^{-1}(165.8)$$

=-4

| | | | | |
|----|-------|----|----|----|
| 20 | 22 | 30 | 40 | 55 |
| 23 | 165.8 | 55 | 20 | 47 |
| 23 | 76 | 28 | 35 | 85 |
| 12 | 46 | 87 | 64 | 63 |
| 92 | 86 | 42 | 54 | 75 |

And all images we take same think

3- Prewitt operator

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

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

The same of sobel but different in just the mask

4- Kirsch compass masks

Is called compass masks because they are define single mask and rotation the masks to the eight major orientations.

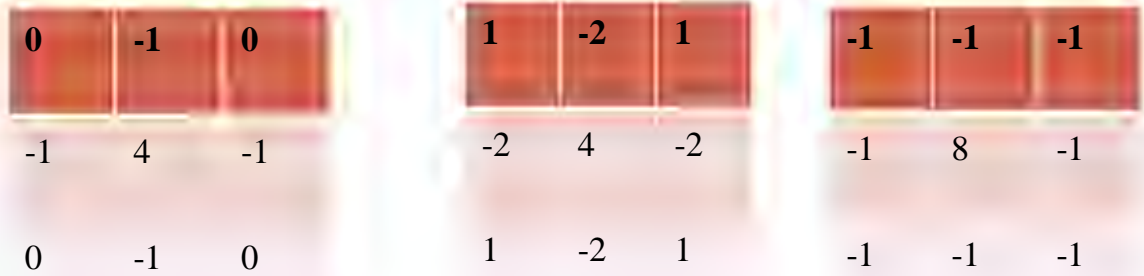
| | | | | | | | | | |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| -3 | -3 | 5 | -3 | 5 | 5 | 5 | 5 | 5 | -3 |
| -3 | 0 | 5 | -3 | -3 | 0 | 5 | -3 | 0 | -3 |
| -3 | -3 | 5 | -3 | -3 | -3 | -3 | -3 | -3 | -3 |
| 5 | -3 | -3 | -3 | -3 | -3 | -3 | -3 | -3 | -3 |
| 5 | 0 | -3 | 5 | 0 | -3 | -3 | 0 | -3 | 5 |
| 5 | -3 | -3 | 5 | 5 | -3 | 5 | 5 | 5 | -3 |

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.

5-Robinson compass masks:

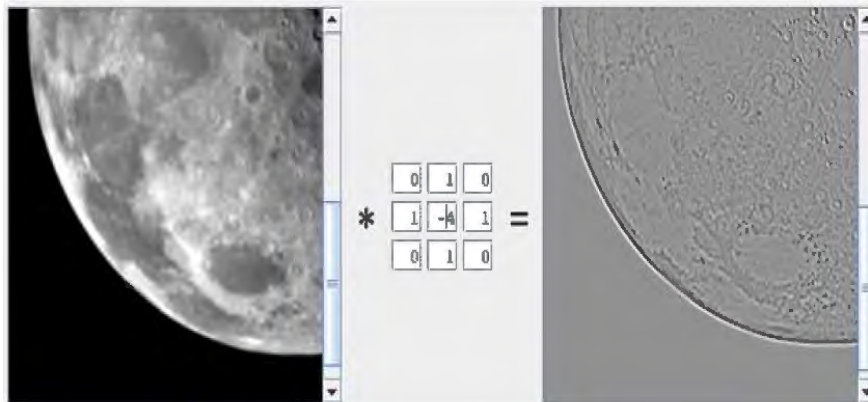
The Robinson compass are used in a manner similar to the kirsch but are easier to implemented because rely only compute in (0,1,2).

6- 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.

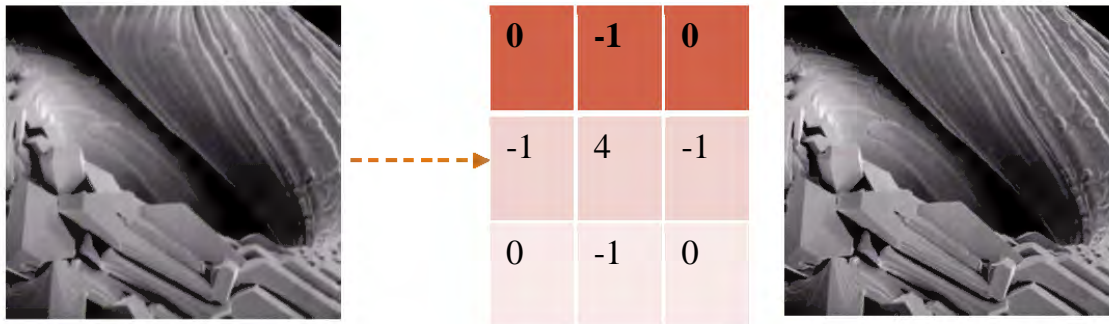


The preceding convolution masks would return a value of zero. If we want to retain most of the information that is in the original the coefficient should sum to a number greater than zero.

For example see bellow



Otherwise we can use the Laplacian in enhancement for example



7- Frei – chen masks:The frei chen is unique in that they form a complete set of basis vector. This mean we can represent any 3*3 sub image as a weighted sum of nine frei chen , this weighted can be found by projection the sum image with all of nine frei chen masks. We have nine masks of frei- chen this masks is

$$\begin{array}{ccc}
 \frac{1}{2\sqrt{2}} \begin{bmatrix} 1, & \sqrt{2}, & 1 \\ 0, & 0, & 0 \\ -1, & -\sqrt{2}, & -1 \end{bmatrix} & \frac{1}{2\sqrt{2}} \begin{bmatrix} 1, & 0, & -1 \\ \sqrt{2}, & 0, & -\sqrt{2} \\ 1, & 0, & -1 \end{bmatrix} & \frac{1}{2\sqrt{2}} \begin{bmatrix} 0, & -1, & \sqrt{2} \\ 1, & 0, & -1 \\ -\sqrt{2}, & 1, & 0 \end{bmatrix} \\
 f_1 & f_2 & f_3 \\
 \\
 \frac{1}{2\sqrt{2}} \begin{bmatrix} \sqrt{2}, & -1, & 0 \\ -1, & 0, & 1 \\ 0, & 1, & -\sqrt{2} \end{bmatrix} & \frac{1}{2} \begin{bmatrix} 0, & 1, & 0 \\ -1, & 0, & -1 \\ 0, & 1, & 0 \end{bmatrix} & \frac{1}{2} \begin{bmatrix} -1, & 0, & 1 \\ 0, & 0, & 0 \\ 1, & 0, & -1 \end{bmatrix} \\
 f_4 & f_5 & f_6 \\
 \\
 \frac{1}{6} \begin{bmatrix} 1, & -2, & 1 \\ -2, & 4, & -2 \\ 1, & -2, & 1 \end{bmatrix} & \frac{1}{6} \begin{bmatrix} -2, & 1, & -2 \\ 1, & 4, & 1 \\ -2, & 1, & -2 \end{bmatrix} & \frac{1}{3} \begin{bmatrix} 1, & 1, & 1 \\ 1, & 1, & 1 \\ 1, & 1, & 1 \end{bmatrix} \\
 f_7 & f_8 & f_9
 \end{array}$$

Example:

With image=

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

And we can make a projection of nine mask on this image to find the edge

For (f1) =

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

$$\frac{1}{2\sqrt{2}} [1(1) + 0(\sqrt{2}) + 1(1) + 1(0) + 0(0) + 1(0) + 1(-1) + 0(-\sqrt{2}) + 1(-1)] = 0$$

We can see that the projection of f1 the result = 0 and when complete the projection of all masks the result is:

$$f1 \rightarrow 0, f2 \rightarrow 0, f3 \rightarrow 0, f4 \rightarrow 0, f5 \rightarrow -1, f6 \rightarrow 0, f7 \rightarrow 0, f8 \rightarrow -1, f9 \rightarrow 2$$

And we take only the weight that not zero in this value only the mask (F5,F8,F9) and multiplication with mask and summation all the result.

$$= -1 * \frac{1}{2} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} + -1 * \frac{1}{6} \begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix} + 2 * \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} = I_s$$

Then used the information to find the edge by

$$\cos \theta = (\sqrt{M/S}) \quad \text{where} \quad M = \sum_{k \in \{e\}} (I_s, F_k)^2 \quad \text{and} \quad S = \sum_{k=1}^9 (I_s, F_k)^2$$

Set $\{e\}$ consists of the masks of interest. The (I_s, Fk) notation refers to the process of overlaying the mask on the sub image, multiplying coincident terms, and summing the result.

3- Edge operator performance:

If we need to developed a performance metric for edge detection we need to define:

1. Missing valid edge point.
2. Classifying noise pulses as valid edge points.
3. Smearing edge.

If this point does not accrue we can say that we have achieved success. From all operators well we have to comparison with all operators, we get example on noise image and show the result of all operators. Other than by masking, edge detection can also be performed by *subtraction*. Two methods that use subtraction to detect the edge are **Homogeneity operator** and **Difference operator**.

The **homogeneity operator** subtracts each of the pixels next to the centre of the $n \times n$ area (where n is usually 3) from the centre 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 **difference operator** performs differentiation by calculating the differences between the pixels that surround the centre pixel of an $n \times n$ 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. But it is quicker than the homogeneity operator since it uses four integer subtractions as against eight subtractions in homogeneity operator per pixel. Shown below is how the two operators detect the edge: Consider an image block with centre pixel intensity 5,

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

Output of *homogeneity operator* is:

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

Output of *difference operator* is:

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

LECTURE 7

Histogram

The histogram of an image is a plot of the gray _levels values versus the number of pixels at that value.

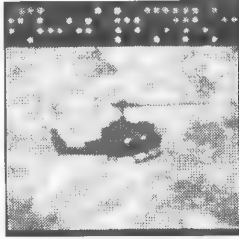
A histogram appears as a graph with "brightness" on the horizontal axis from 0 to 255 (for an 8-bit) intensity scale) and "number of pixels "on the vertical axis. For each colored image three histogram are computed, one for each component (RGB, HSL).The histogram gives us a convenient -easy -to -read representation of the concentration of pixels versus brightness of an image, using this graph we able to see immediately:

- 1 Whether an image is basically dark or light and high or low contrast.
- 2 Give us our first clues about what contrast enhancement would be appropriately applied to make the image more subjectively pleasing to an observer, or easier to interpret by succeeding image analysis operations.

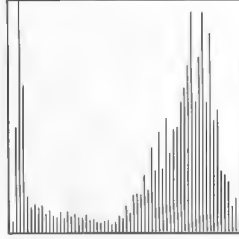
So the shape of histogram provide us with information about nature of the image or sub image if we considering an object within the image. For example:

- 1 Very narrow histogram implies a low-contrast image.
- 2 Histogram skewed to word the high end implies a bright image.
- 3 Histogram with two major peaks , called bimodal, implies an object that is in contrast with the background.

Examples of the different types of histograms are shown in figure below.



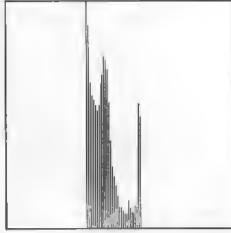
a. Object in contrast with background.



b. Histogram of (a) shows bimodal shape.



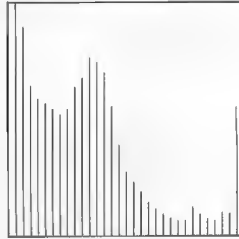
c. Low-contrast image.



d. Histogram of (c) appears clustered.



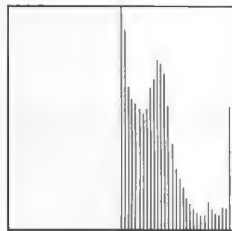
e. High-contrast image.



f. Histogram of (e) appears spread out.



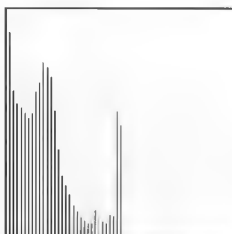
g. Bright image.



h. Histogram of (g) appears shifted to the right.



i. Dark image.

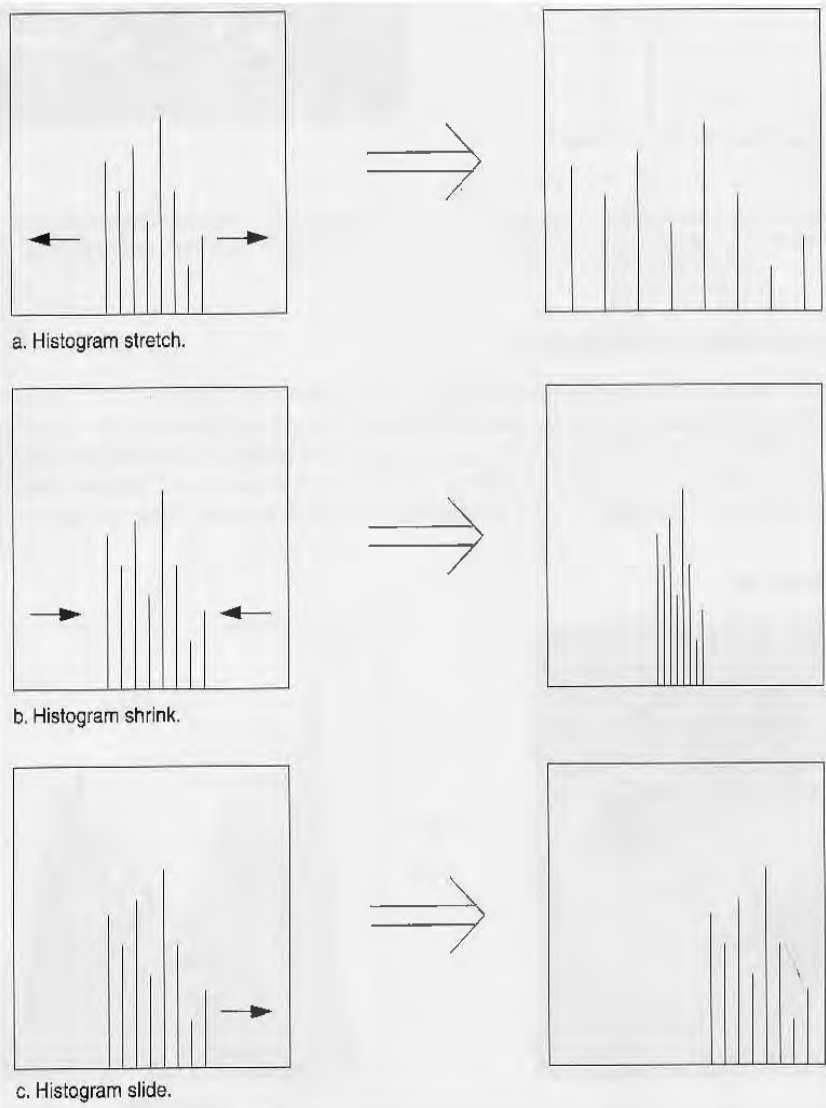


i. Histogram of (i) appears shifted to

Different types of Histogram

Histogram Modifications

The gray level histogram of an image is the distribution of the gray level in an image. The histogram can be modified by mapping functions, which will stretch, shrink (compress), or slide the histogram. Figure below illustrates a graphical representation of histogram stretch, shrink and slide.



Histogram Modifications.

- The mapping function for histogram stretch can be found by the following equation:

$$\text{Stretch}(I(r, c)) = \left[\frac{I(r, c) - I(r, c)_{\min}}{I(r, c)_{\max} - I(r, c)_{\min}} \right] [MAX - MIN] + MIN.$$

Where, $I(r, c)_{\max}$ is the largest gray-level in the image $I(r, c)$.

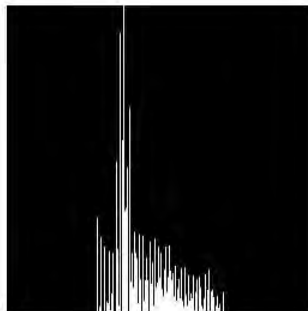
$I(r, c)_{\min}$ is the smallest gray-level in the image $I(r, c)$.

MAX and MIN correspond to the maximum and minimum gray-level values possible (for an 8-bit image these are 255 and 0).

This equation will take an image and stretch the histogram across the entire gray-level range which has the effect of increasing the contrast of a low contrast image (see figure below of histogram stretching).



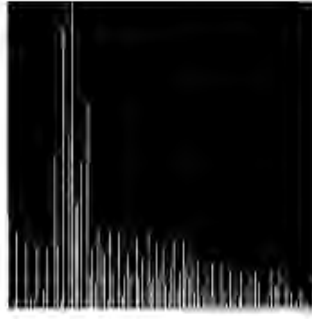
Low-contrast image



Histogram of low-contrast image



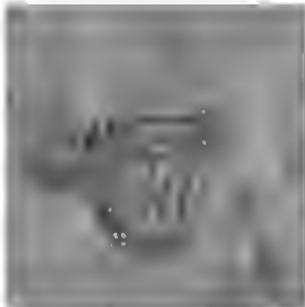
Image after histogram stretching



Histogram of image after stretching

Histogram Stretching.

In most of the pixel values in an image fall within small range, but a few outlines force the histogram to span the entire range, a pure histogram stretch will not improve the image. In this case it is useful to allow a small proceeding of the pixel values to be clipped at the low and high end of the range (for an 8-bit image this means truncating at 0 and 255). See figure below of stretched and clipped histogram.



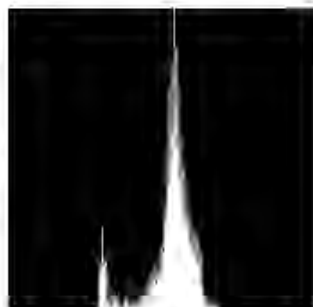
Original Image



Histogram of the original image



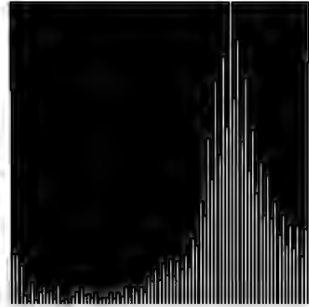
Image after histogram stretching without clipping



Histogram of the image



Image after histogram stretching with clipping 3% low and high value



Histogram of the image

Histogram Stretching (Clipping).

- The opposite of a histogram stretch is a histogram shrink, which will decrease image contrast by compressing the gray levels. The mapping function for a histogram shrinking can be found by the following equation:

$$\text{Shrink}(I(r,c)) = \left[\frac{\text{Shrink}_{\max} - \text{Shrink}_{\min}}{I(r,c)_{\max} - I(r,c)_{\min}} \right] [I(r,c) - I(r,c)_{\min}] + \text{Shrink}_{\min}$$

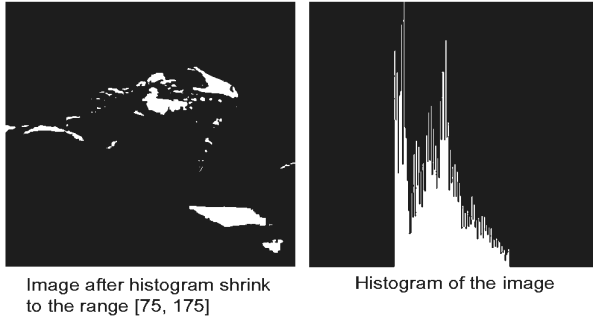
Shrink_{\max} and shrink_{\min} correspond to the maximum and minimum desired in the compressed histogram. In general, this process produces an image of reduced contrast and may not seem to be useful an image enhancement (see figure below of shrink histogram).



Original image



Histogram of original image



Histogram Shrinking.

- The histogram slide techniques can be used to make an image either darker or lighter but retain the relationship between gray-level values. This can be accomplished by simply adding or subtracting a fixed number for all the gray-level values, as follows:

$$\text{Slide } (I(r,c)) = I(r,c) + \text{OFFSET.}$$

Where OFFSET values is the amount to slide the histogram.

In this equation, a positive OFFSET value will increase the overall brightness; where as a negative OFFSET will create a darker image, figure below shows histogram sliding.

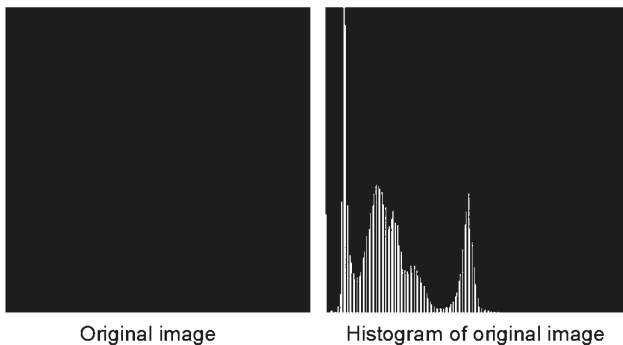




Image after positive-value histogram sliding



Histogram of image after sliding

Histogram Sliding.

Histogram Equalization

Is a popular technique for improving the appearance of a poor image. It's a function similar to that of a histogram stretch but often provides more visually pleasing results across a wide range of images.

Histogram equalization is a technique where the histogram of the resultant image is as flat as possible (with histogram stretching the overall shape of the histogram remains the same).

The results in a histogram with a mountain grouped closely together to "spreading or flattening histogram makes the dark pixels appear darker and the light pixels appear lighter.

The histogram equalization process for digital images consists of four steps:

1. Find the running sum of the histogram values
2. Normalize the values from step1 by dividing by total number of pixels.
3. Multiply the values from step2 by the maximum gray level value and round.
4. Map the gray-level values to the results from step 3, using a one-to-one correspondence. The following example will help to clarify this process.

Example:-

We have an image with 3 bit /pixel, so the possible range of values is 0 to 7. We have an image with the following histogram:

| | | | | | | | | |
|------------------|----|---|---|---|----|---|---|---|
| Gray-level value | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| No of Pixel | 10 | 8 | 9 | 2 | 14 | 1 | 5 | 2 |
| Histogram value | | | | | | | | |

Step 1: Create a running sum of histogram values. This means that the first values is 10, the second is $10+8=18$, next is $10+8+9=27$, and soon. Here we get 10,18,29,43,44,49,51.

Step 2: Normalize by dividing by total number of pixels. The total number of pixels is $10+8+9+2+14+1+5+2=51$.

Step 3 : Multiply these values by the maximum gray – level values in this case 7 , and then round the result to the closet integer. After this is done we obtain 1,2,4,4,6,6,7,7.

Step 4 : Map the original values to the results from step3 by a one –to-one correspondence.

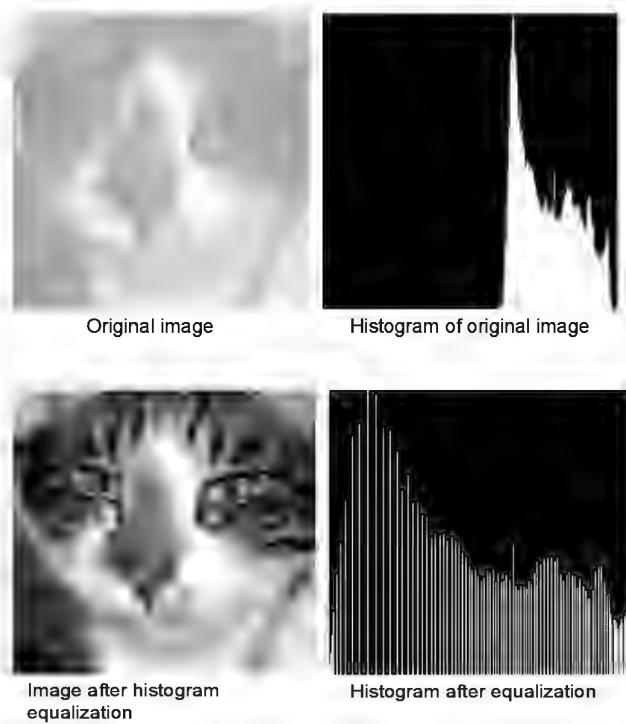
The first three steps:

| | | | | | | | | |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Gray-level | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| No. of Pixel | 10 | 8 | 9 | 2 | 14 | 1 | 5 | 2 |
| Run Sum | 10 | 18 | 27 | 29 | 43 | 44 | 49 | 51 |
| Normalized | 10/51 | 18/51 | 27/51 | 29/51 | 43/51 | 44/51 | 49/51 | 51/51 |
| Multiply by 7 | 1 | 2 | 4 | 4 | 6 | 6 | 7 | 7 |

The fourth step:

| | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|
| Old | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| New | 1 | 2 | 4 | 4 | 6 | 6 | 7 | 7 |

All pixel in the original image with gray level 0 are set to 1, values of 1 are set to 2, 2 set to 4, 3 set to 4, and so on (see figure below) histogram equalization, you can see the original histogram and the resulting histogram equalized histogram. Although the result is not flat, it is closer to being flat than the original.



Histogram Equalization.