1. Introduction

Machine Vision

Computer and machine vision involve the automatic extraction, manipulation analysis and classification of images or image sequence, usually within special or general-purpose computering systems. The purpose of which is to obtain useful information about the word with a view to carrying out some task.

Many researchers use the terms computer and machine vision interchangeably, although there are key differences. Machine vision systems are generally used in an industrial environment. The design of machine vision systems requires a broad spectrum of techniques and disciplines. These include electronic engineering (hardware and software design), physics (optics and lighting), mechanical engineering (since industrial vision systems deal with a mainly mechanical world) as well as the system engineering aspects of developing reliable industrial systems. Machine vision generally involves the engineering of solutions to specific problems and can be seen as a subset of the general system engineering task.

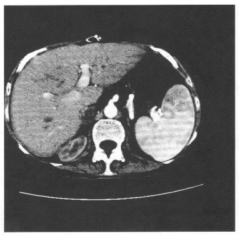
The formal definition of machine vision put forward by the Automated Vision Association (AVA)refers to it as :

"The use of devices for optical ,non-contact sensing to automatically receive and interpret an image of real scene in order to obtain information and/or control machines or processes". The main application area for machine vision occurs in robotics.

The goal of a machine vision system is to create a model of the real world from images. A machine vision system recovers useful information about a scene from its two-dimensional projections. Since images are two-dimensional projections of the three-dimensional world, the information is not directly available and must be recovered. To recover the information, knowledge about the objects in the scene and projection geometry is required. In the table below we see the relations between image processing and computer vision.

	Output		
Input		Image	Description
	Image	Image processing	Computer vision
	Description	Computer graphics	Other data processing

The below images show haw the machines used to have an images then converted to image data.



Medical images using computed tomography images

Image acquire from mobile robot

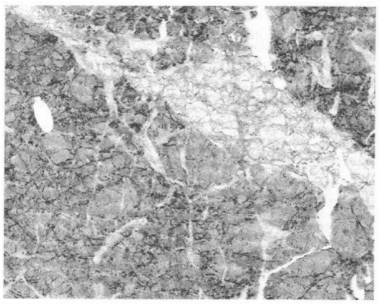


Image of an arctic region taken from a satellite

Computer imaging:

Can be defined as acquisition and processing of visual information by computer. Computer imaging can be separate into two primary categories: computer vision and image processing.

Computer Vision

It is a 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.

Digital image analysis is related to the description and recognition of the digital content. Its input is a digital image and its output is a symbolic image

description. In many cases, digital image analysis techniques simulate human vision functions. Therefore, the term computer vision can be used as equivalent to (or the superset of) digital image analysis.

Computer vision is a branch of computer science which concentrates on the wider scientific issues relating to the automated interpretation of images. Computer vision often deals with relatively domain-independent considerations, generally aiming to duplicate the effect of human vision by electronically perceiving and understanding an image. Research in this area often deals with issues involving perception psychology and the analysis of human and other biological vision system. One of the most commonly used applications of computer vision is the pattern recognition.

Computer vision systems are used in many and various types of environments, such as manufacturing systems, medical community, infrared imaging, and satellites orbiting.

Image Processing:

Image processing is a 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.

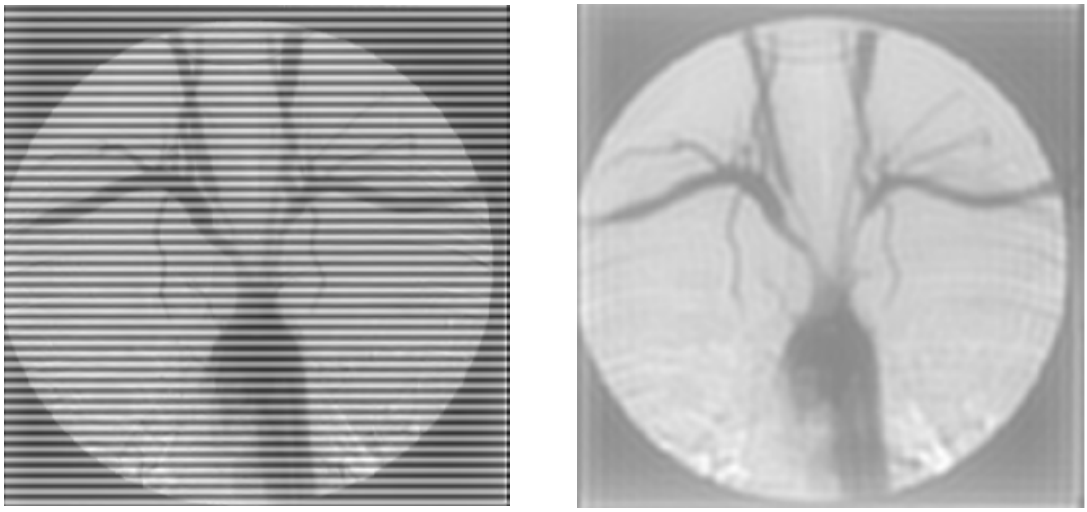
The major topics within the field of image processing include:

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

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**

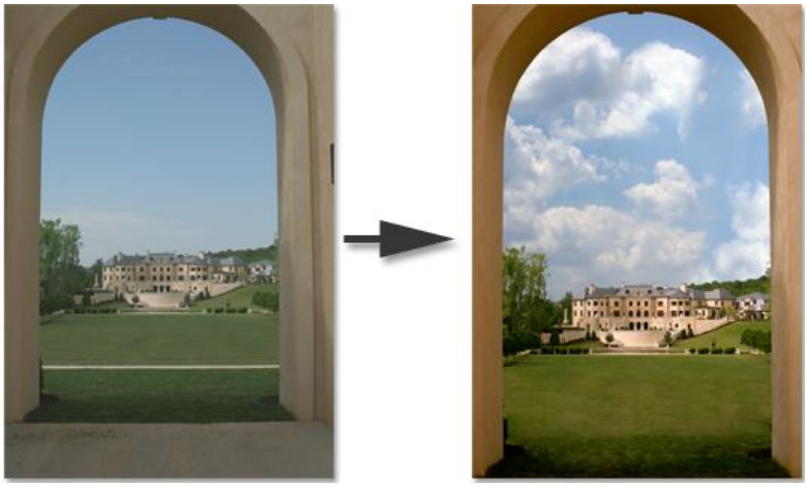
Image Restoration

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. They differ in how they approach the problem. 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**

Image Enhancement

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 processing systems are used in many and various types of environments, such as: Medical community , Computer – Aided Design , Virtual Reality, Image Processing.

a. Image before compression (92) KB **b. Image after compression (6.59)KB**

Image Compression.

Image Representation

We have seen that the human visual system (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. 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 (يمسك) an object or 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').

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.

Gray Scale Images.

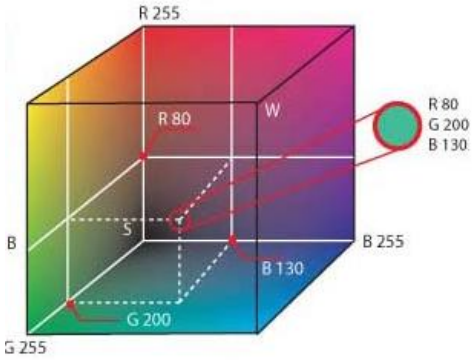
3. Color Image

Color image can be modeled as three band monochrome image data, where each band of the data corresponds to a different color.

Color Images

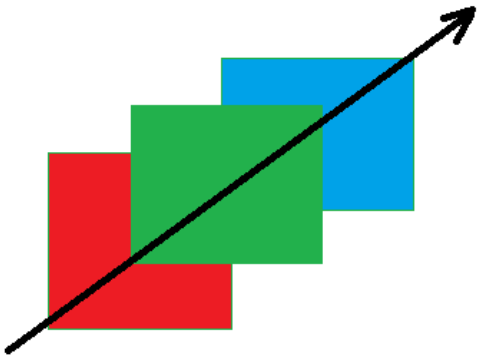
The actual information stored in the digital image data is brightness information in each spectral band. When the image is displayed, the corresponding brightness information is displayed on the screen by picture elements that emit light energy corresponding to that particular color.

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.



RGB color image

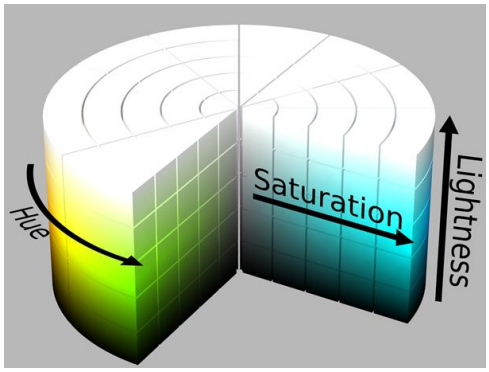
The following figure illustrate that in addition to referring to arrow or column as a vector, we can refer to a single pixel red ,green, and blue values as a color pixel vector –(R,G,B).



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 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.



HSL Color Space[1].

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].

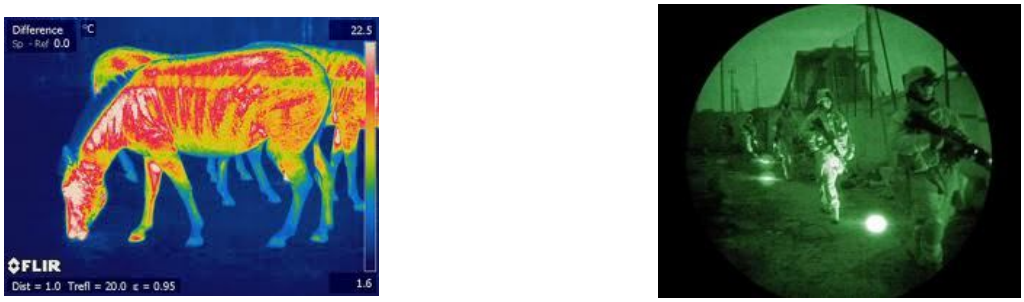
Example: "a deep, bright orange" would have a large intensity ("bright"), a hue of "orange", and a high value of saturation ("deep").we can picture this color in our minds, but if we defined this color in terms of its RGB components, R=245, G=110 and B=20.

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

4. Multispectral Images

Multispecrtal 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.



Multispectral images

2. Image Analysis

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 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.

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. 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.

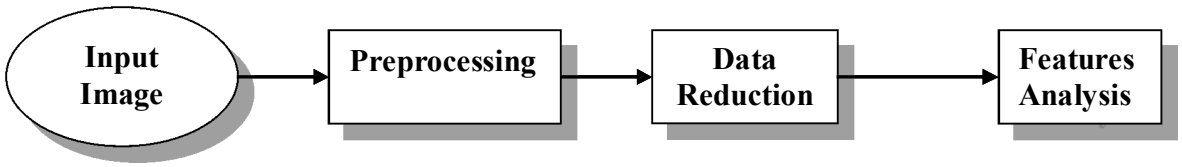
2. Data Reduction:

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.

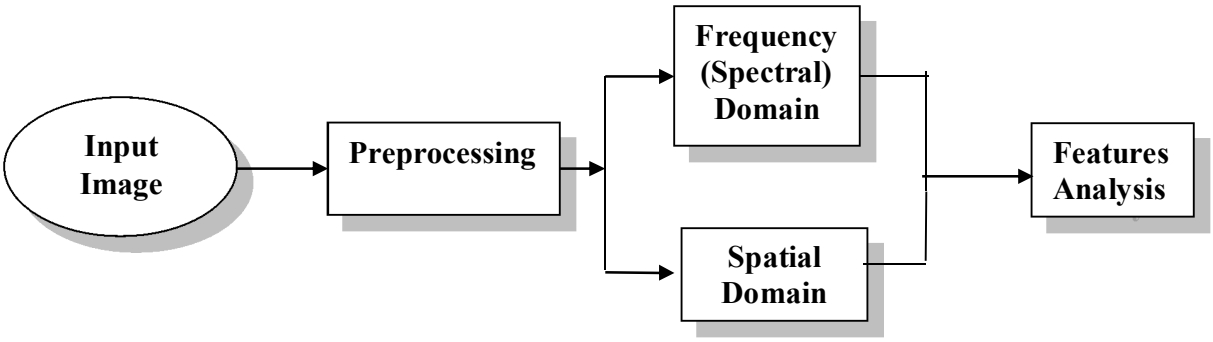
3. Features Analysis:

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

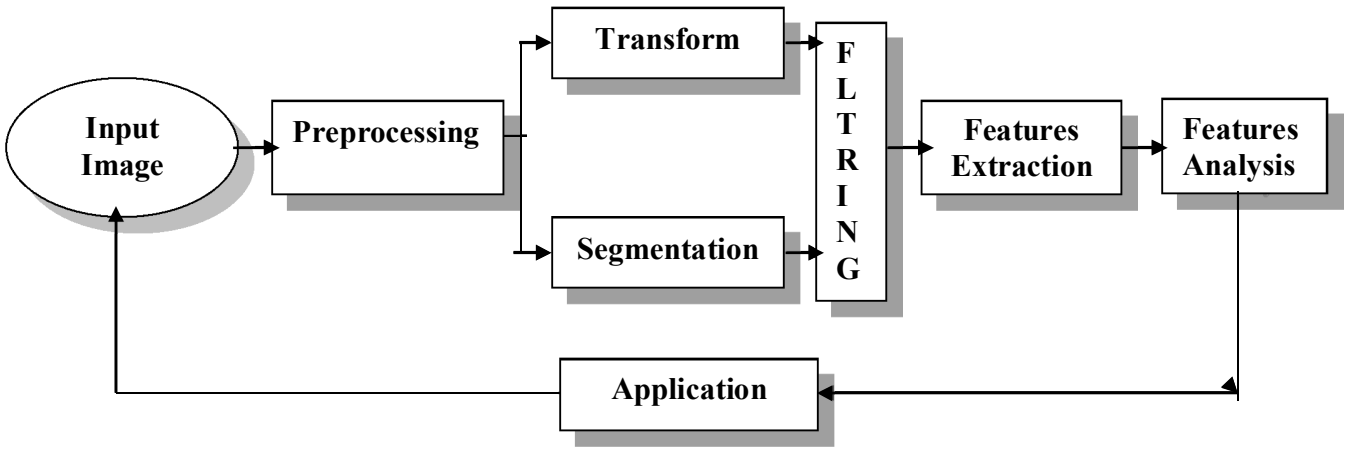
After preprocessing we can perform segmentation on the image in the spatial domain or convert it into the frequency domain via a mathematical transform. After these processes we may choose to filter the image. This filtering process further reduces the data and allows us to extract the feature for analysis.



a. Image Analysis



b. Image Analysis Domains



c. Image Analysis

Preprocessing

The preprocessing algorithm, techniques and operators are use 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.

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:

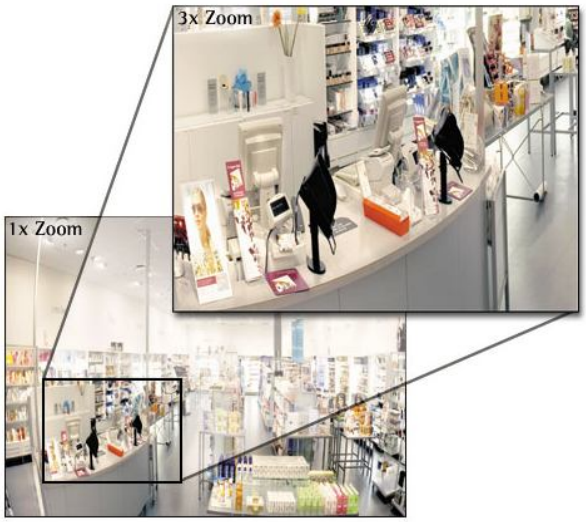
Crop , Zoom, enlarge , shrink, translate and rotate.

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. **Zero-Order Hold.**
2. **First _Order Hold.**
3. **Convolution.**

1. Zero-Order hold: is performed by repeating previous pixel values, thus creating a blocky effect as in the following figure:



Zero _Order Hold Method

2. **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

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

Image with Rows Expanded

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

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

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

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

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

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

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

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

The convolution process requires us to overlay the mask on the image, multiply the coincident (متقابله) 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

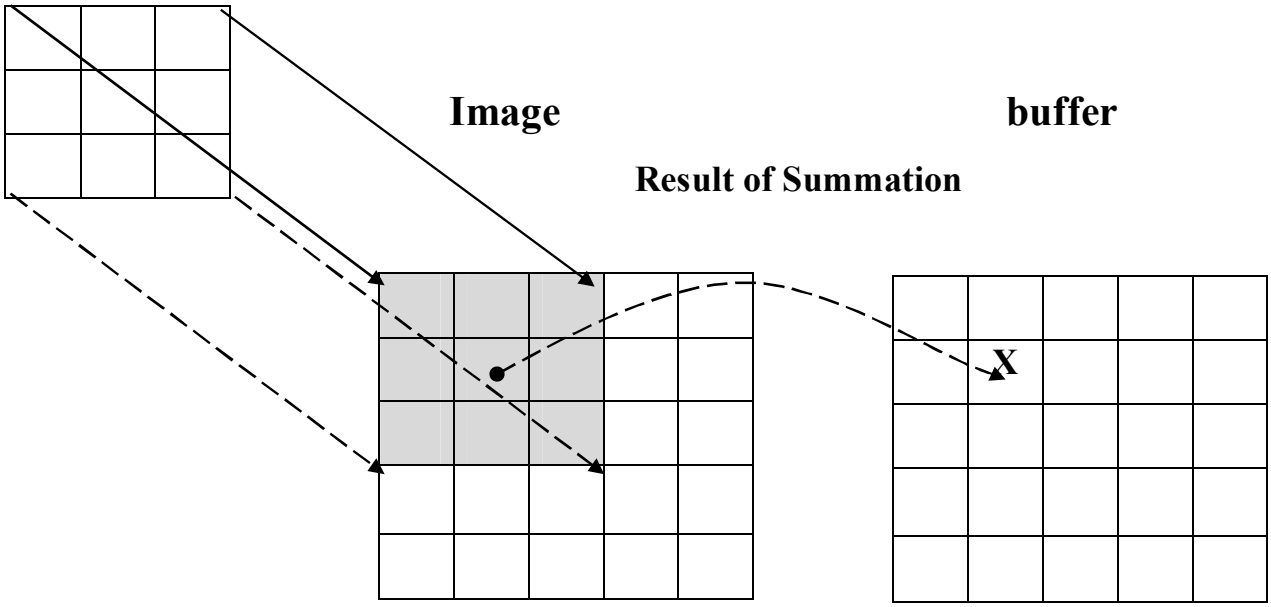
the entire image, the process of sliding, multiplying and summing is called convolution.

First _Order Hold Method

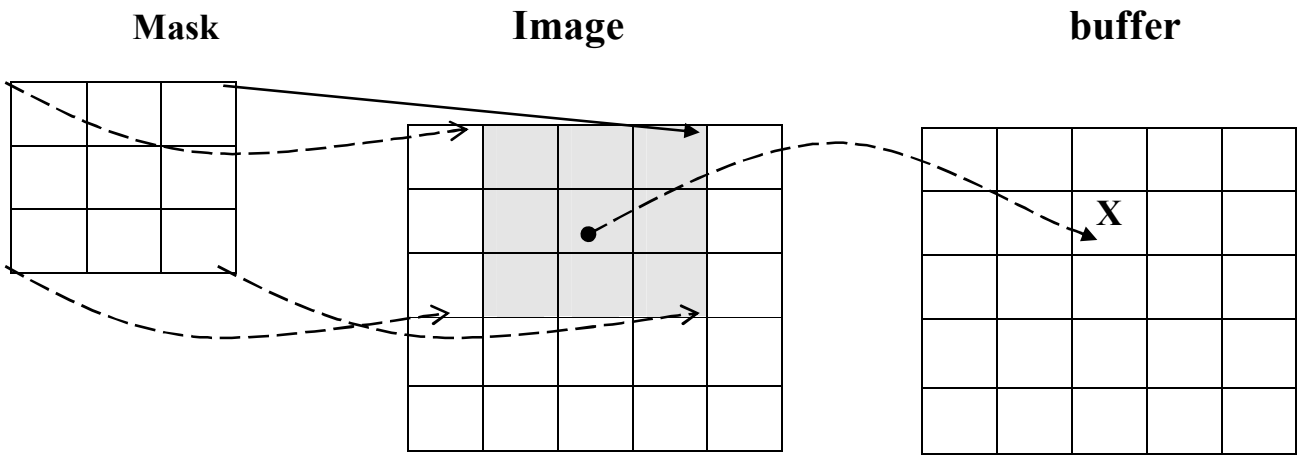
Note that the output image must be put in a separate image array called a buffer, so that the existing values are not overwritten during the convolution process.

The convolution process

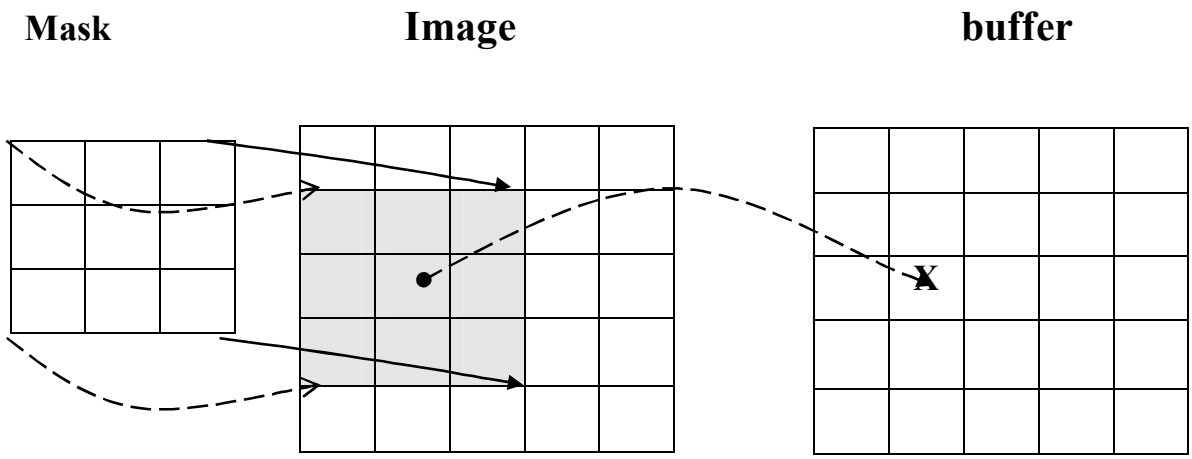
Mask



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).



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.



c. Move the mask down on 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).

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. Note, only first-order hold be performed via convolution, but zero-order hold can also achieved by extending the image with zeros and using the following convolution mask.

Zero-order hold convolution mask

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

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.

These methods will only allows 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: we want to enlarge an image to three times its original size, and we have two adjacent pixel values 125 and 140.

1. Find the difference between the two values, 140-125 =15.
2. The desired enlargement is k=3, so we get 15/3=5.
3. next determine how many intermediate pixel values .we need :
   K-1=3-1=2. The two pixel values between the 125 and 140 are
   125+5=130 and 125+2*5 = 135.

- We do this for every pair of adjacent pixels .first along the rows and then along the columns. This will allows us to enlarge the image by any factor of K (N-1) +1 where K is an integer and N×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 easer to see.

3. Image Analysis

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 I_1 and I_2 to create I_3:

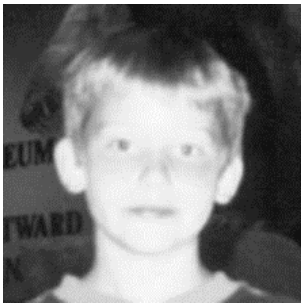
$$\underset{\mathbf{I_1}}{\begin{pmatrix} 3 & 4 & 7 \\ 3 & 4 & 5 \\ 2 & 4 & 6 \end{pmatrix}} + \underset{\mathbf{I_2}}{\begin{pmatrix} 6 & 6 & 6 \\ 4 & 2 & 6 \\ 3 & 5 & 5 \end{pmatrix}} = \underset{\mathbf{I_3}}{\begin{pmatrix} 3+6 & 4+6 & 7+6 \\ 3+4 & 4+2 & 5+6 \\ 2+3 & 4+5 & 6+5 \end{pmatrix}} = \begin{pmatrix}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

- 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.

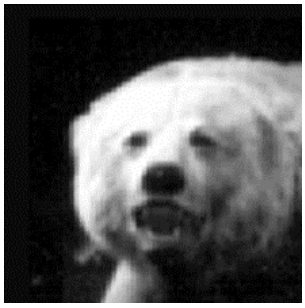
If something has moved in the scene, subtraction produces a nonzero result at the location of movement. Applications include Object tracking , Medical imaging, Law enforcement and Military applications

- 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 grater 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.



a. First Original image



b. Second Original

c. Addition of two images

Image Addition.

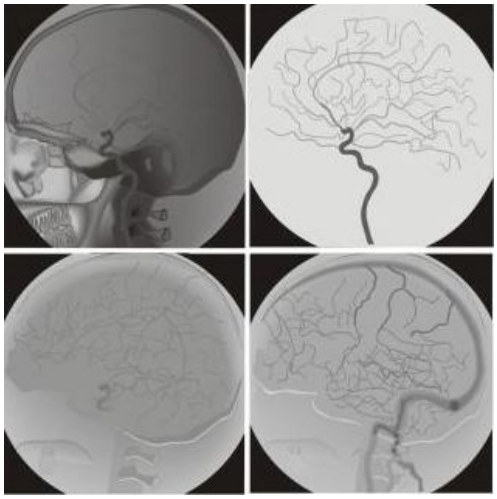


Image Subtraction.

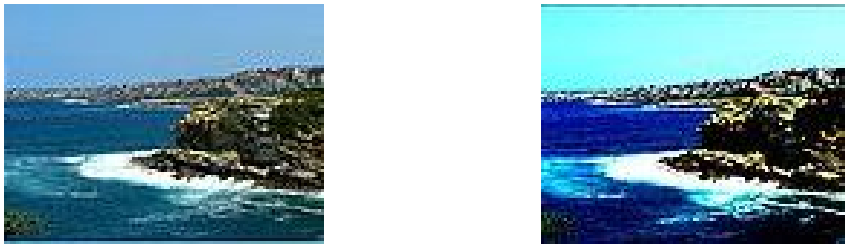


Image Multiplication.

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:

$(111)_{10}$ ⟶ 01101111_2

AND

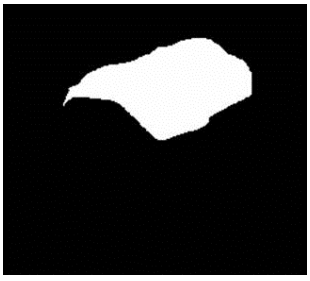
$(88)_{10}$ ⟶ 01011000_2

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 asimple method to extract a Region of Interest from an image, if more sophisticated graphical methods are not available.

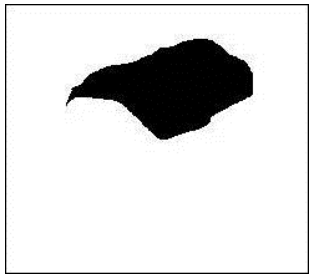
Example: A white square ANDed with an image will allow only the portion of the image coincident with the square to appear in the output image with the background turned black; and a black square ORd with an image will allow only the part of the image corresponding to the black square to appear in the 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.

a. Original image

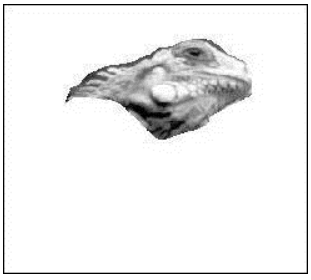


b. Image mask (AND)

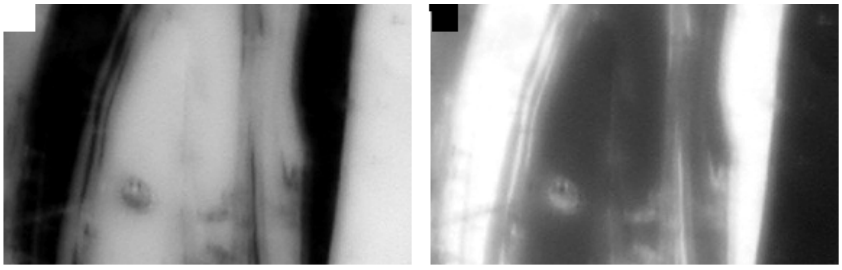
c. ANDing a and b



d. Image mask (OR)



e. ORing a and d



Complement Image.

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.

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.

What is 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.

Image Noise.

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 neighbours, 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 neighbourhoods and replace the centre pixel with an average of the pixels in this neighbourhood. This replacement is done with a convolution mask such as the following 3X3 mask

Arithmetic mean filter smoothing or low-pass filter.

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

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 smoothes 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.

The median filter 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.

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

Where $I_1, I_2, I_3 \ldots\ldots\ldots, I_N$ are the intensity values of the subset of pixels in the image.


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 , un this case it is 5. This 5 is then placed in centre location.

A median filter can use a neighbourhood of any size, but 3X3, 5X5 and 7X7 are typical. Note that the output image must be written to a separate image (a 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 window is overlaid on the upper left corner of the image, and the median is determined. This value is put into the output image (buffer) corresponding to the centre location of the window. The window is then slide one pixel over, and the process is repeated.

When the end of the row is reached, the window is slide back to the left side of the image and down one row, and the process is repeated. This process continues until the entire image has been processed.

Note that the outer rows and columns 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). For example, with 3X3 mask, we lose one outer row and column, a 5X5 mask we lose two rows and columns. This is not visually significant for a typical 256X256 or 512X512 images.

The maximum and minimum filters are two order filters that can be used for elimination of salt- and-pepper noise. The maximum filter selects the largest value within an ordered window of pixels values; where as the minimum filter selects the smallest value.

The minimum filters works best for salt- type noise (High value), and the maximum filters work best for pepper-type noise.

In a manner similar to the median, minimum and maximum filter, order filter can be defined to select a specific pixel rank within the ordered set. For example we may find for certain type of pepper noise that selecting the second highest values works better than selecting the maximum value. This type of ordered selection is very sensitive to their type of images and their use it is application specific. It should note that, in general a minimum or low rank filter will tend to darken an image and a maximum or high rank filter will tend to brighten an image.

The midpoint filter is actually both order and mean filter because it rely on ordering the pixel values , but then calculated by an averaging process. This midpoint filter is the average of the maximum and minimum within the window as follows:

$$\text{Order set} = \mathbf{I_1 \le I_2 \le I_3 \ldots\ldots\ldots\ldots\ldots\ldots \le I_{N^2}}.$$

$$\mathbf{Midpoint = (I_1 + I_{N^2})/2}$$

The midpoint filter is most useful for Gaussian and uniform noise.

The Enhancement filter:

The enhancement filters are:

1. Laplacian type.
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:

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix} \quad \begin{pmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{pmatrix} \quad \begin{pmatrix} -2 & 1 & -2 \\ 1 & 5 & 1 \\ -2 & 1 & -2 \end{pmatrix}$$

The Laplacian type filters will enhance details in all directions equally.

a. Original image

b. Laplacian filtered image

Laplacian Filter.

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.

Vertical

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

Horizontal

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

Diagonal 1

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

Diagonal 2

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

a. Original image

b. Difference filtered image

Difference Filter

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 every thing above that value equal to "1" and every thing below the threshold equal to "0". This effectively turns a gray_level image into abinary (two_level) image and is often used as apreprocessing 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:

We want to reduce 8_bit information containing 256 possible gray_level values down to 32 possible values.

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 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 level, 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 method.

Original 8-bit image,
256 gray levels

Quantized to 6 bits,
64 gray levels

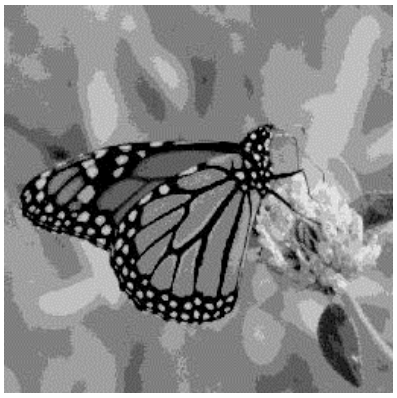
Quantized to 3 bits,
8 gray levels

Quantized to 1 bits,
2 gray levels

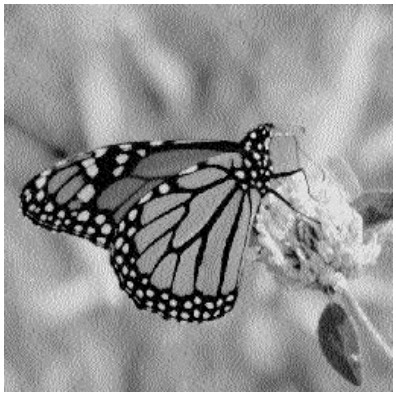
False Contouring

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)

IGS quantization

4. Image Classification

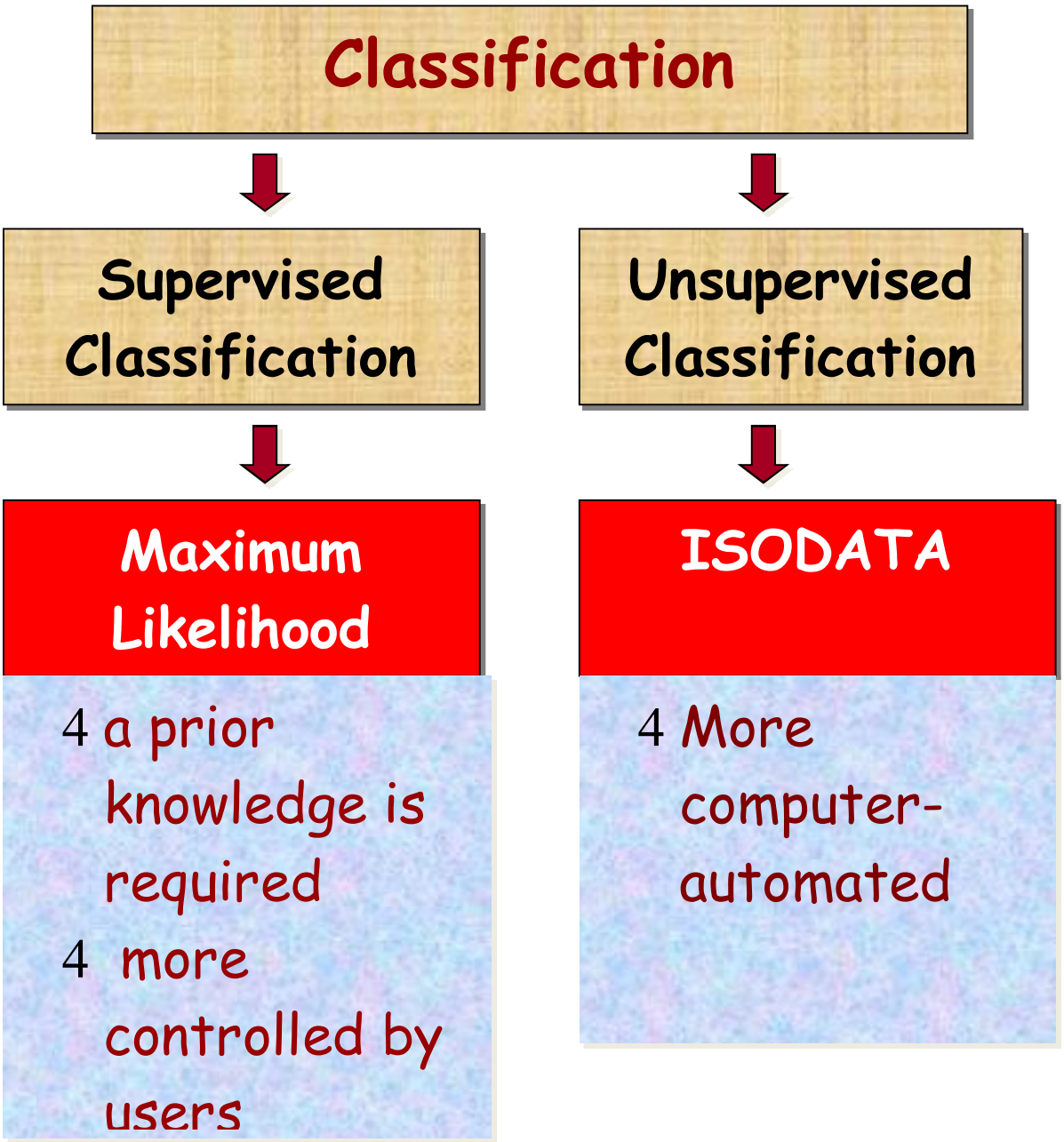
Image classification is an important part of the remote sensing, image analysis and pattern recognition. In some instances, the classification itself may be the object of the analysis. The image classification therefore forms an important tool for examination of the digital images.

The term classifier refers loosely to a computer program that implements a specific procedure for image classification. The analyst must select a classification method that will best accomplish a specific task. At present, it is not possible to state which classifier is best for all situation as the characteristic of each image and the circumstances for each study vary so greatly. Therefore, it is essential that each analyst understand the alternative strategies for image classification so that he or she may be prepared to select the most appropriate classifier for the task in hand.

At present, there is different image classification procedures used for different purposes by various researchers. These techniques are distinguished in two main ways as:

1. Supervised classification
   - Analyst identifies representative training sets for each informational class
   - Algorithm generates decision boundaries
2. Unsupervised classification
   - Algorithm identifies clusters in data
   - Analyst labels clusters

Additionally, supervised classification has different sub classification methods which are named as parallel piped, maximum likelihood, minimum distances and Fisher classifier methods. These methods are named as Hard Classifier.



Maximum likelihood classification algorithm

- The *maximum likelihood decision rule* is based on *probability*.
- It assigns each pixel having pattern measurements or features *X* to the class *i* whose units are most probable or likely to have given rise to feature vector *X*.
- In other words, the probability of a pixel belonging to each of a predefined set of *m* classes is calculated, and the pixel is then assigned to the class for which the probability is the highest.
- The *maximum likelihood decision rule* is one of the most widely used supervised classification algorithms.

The maximum likelihood procedure assumes that the training data statistics for each class in each band are *normally distributed* (Gaussian).

Unsupervised classification (Clustering)

- Only some parameters are required to specify from the user to begin this process.
- Then the computer uses these parameters to uncover statistical patterns that are inherent in the data.
- Spectral classes do not necessarily correspond to any meaning characteristics of ground objects.
- After classification, the users must attach the actual meaning to the resulting classes.

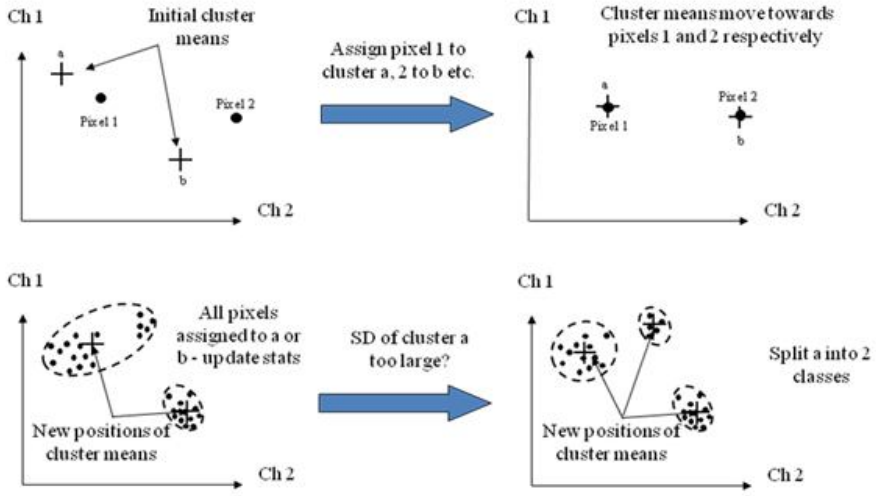
Iterative Self-Organizing Data Analysis Technique (ISODATA)

- *ISODATA* is self-organizing because it requires relatively little human input.
- Don't need to know the number of clusters.
- Algorithm splits and merges clusters.
- User defines threshold values for parameters.
- Computer runs algorithm through many iterations until threshold is reached.

Unsupervised classification: ISODATA algorithm

1. Start with (user-defined number) randomly located clusters (mean values).
2. Assign each pixel to nearest cluster (minimum distance).
3. Re-calculate cluster means and standard deviations.
4. If distance between two clusters < some threshold, merge them.
5. If standard deviation in any one dimension > some threshold, split into two clusters.
6. Delete clusters with small number of pixels.
7. Re-assign pixels, re-calculate cluster statistics etc. until changes of clusters < some fixed threshold.

ISODATA example: 2 classes, 2 bands

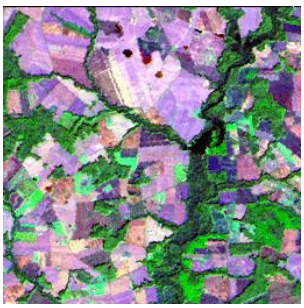


Drawbacks of ISODATA

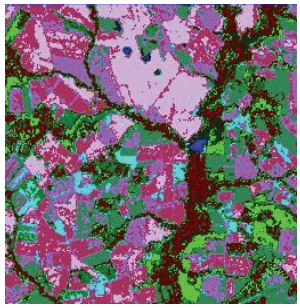
- May be time consuming if data is much unstructured.
- Algorithm can spiral out of control leaving only one class.

Advantages of ISODATA

- Don't need to know much about the data beforehand.
- Little user effort required.
- ISODATA is very effective at identifying spectral clusters in data.



Original Land sat Image



Classified Image

Due to their digital format, the results of digital image classification provide distinct advantages that may make some amount of error tolerable. These advantages allow the results to be

1. Readily provided in hardcopy map form;
2. Compiled in tabular form to provide area, perimeter, and proximity information (such as edge relationships) for each class; and
3 Entered into a geographical information system for subsequent merging and joint analysis with other spatially formatted data.

5. Histogram

Histogram

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

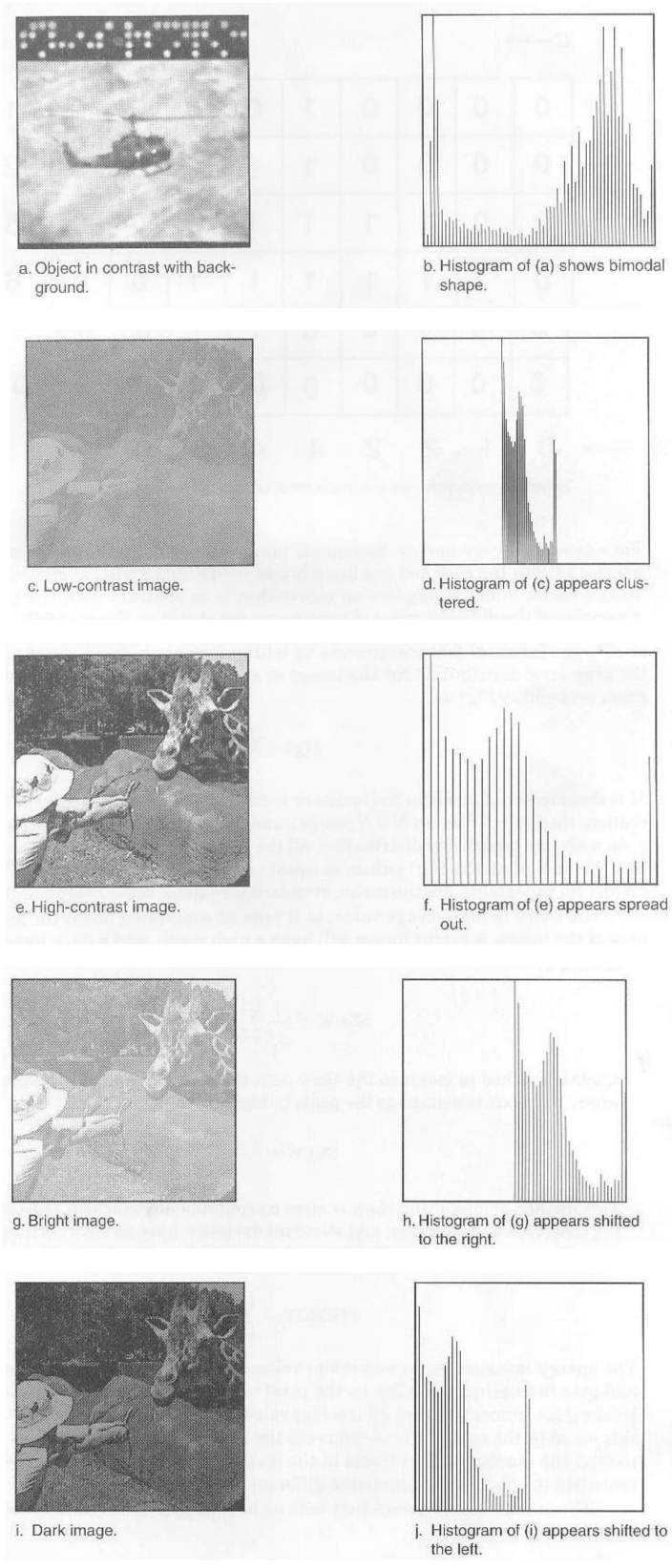
A histogram appears as a graph with "brightness" on the horizontal axis from 0 to 255 (for an 8-bit) intensity scale) and "number of pixels "on the vertical axis. For each colored image three histogram are computed, one for each component (RGB, HSL).The histogram gives us a convenient -easy -to -read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

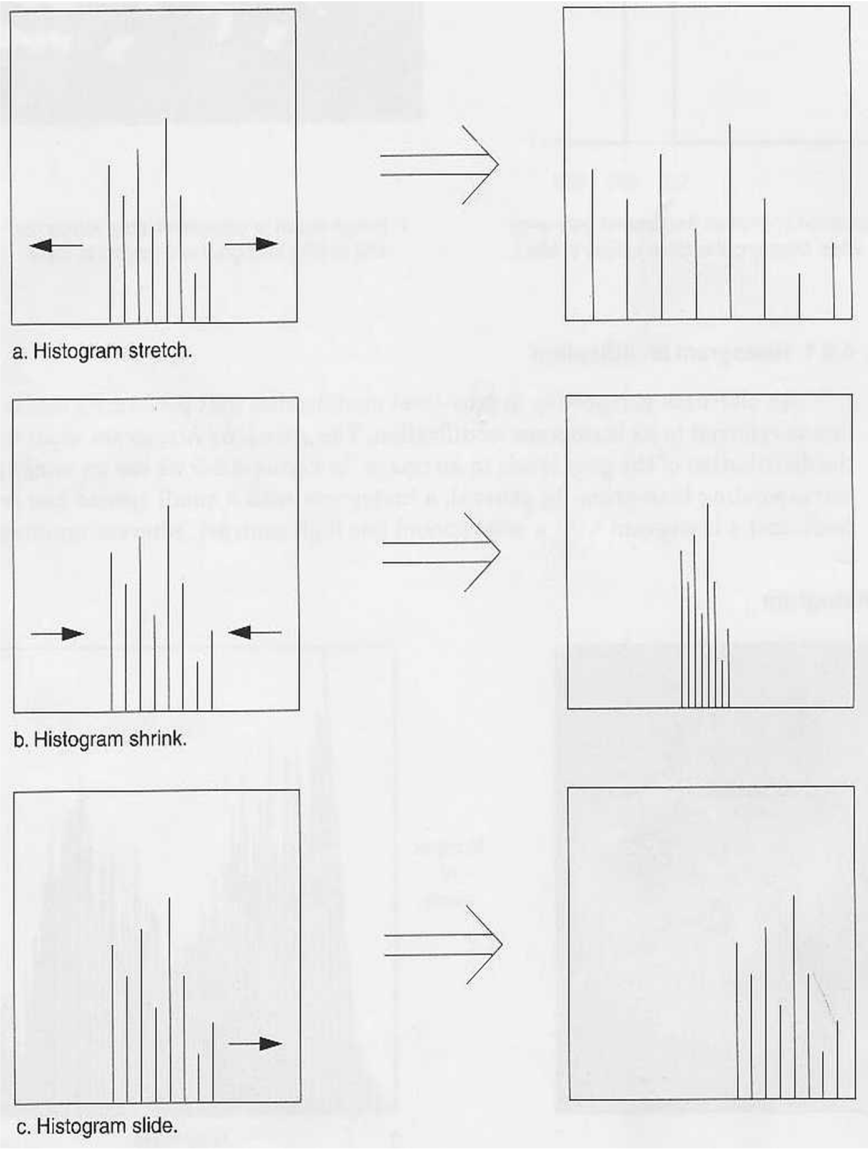
Examples of the different types of histograms are shown in figure below.



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:

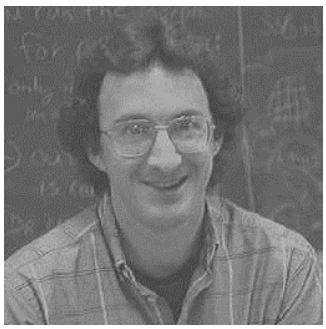
$$\text{Stretch (I (r, c))} = \left[\frac{I(r,c) - I(r,c)_{\min}}{I(r,c)_{\max} - I(r,c)_{\min}} \right] [MAX\text{-}MIN] + MIN.$$

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

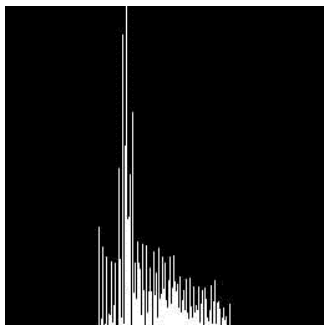
$I(r,c)_{min}$ is the smallest gray- level in the image I(r,c).

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

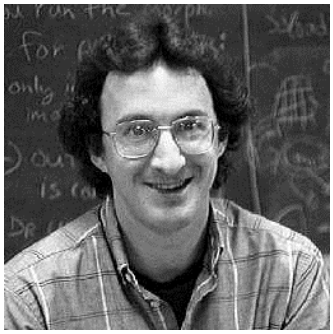
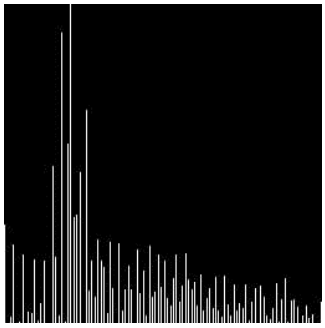
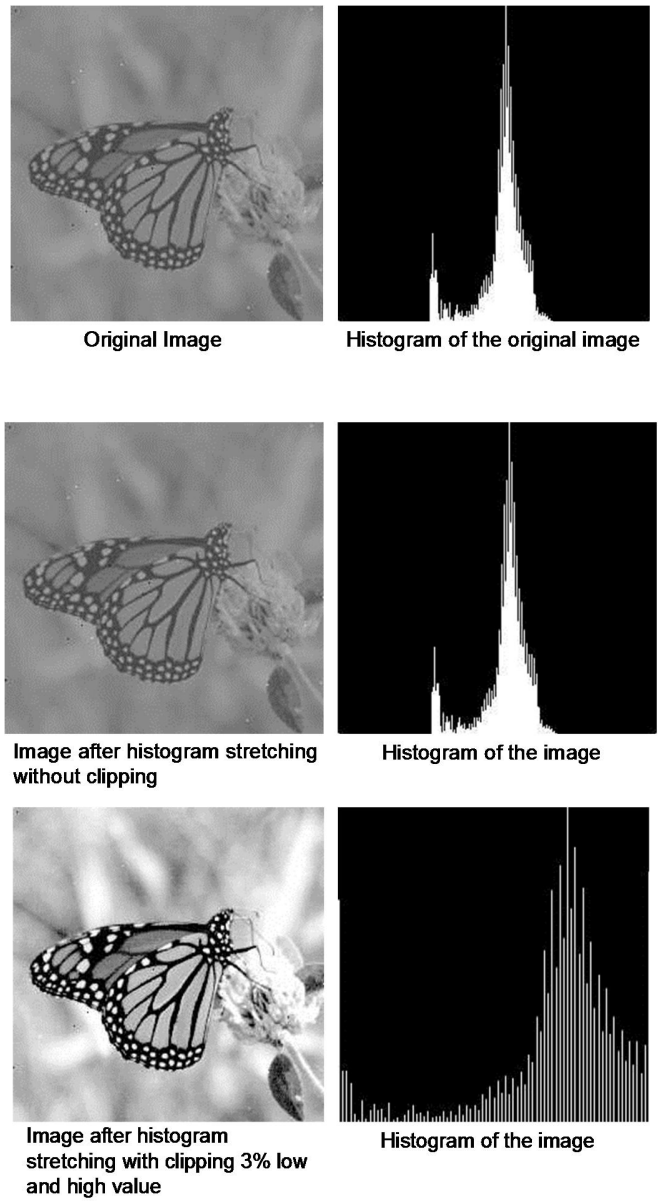


Image after histogram stretching



Histogram of image after 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.



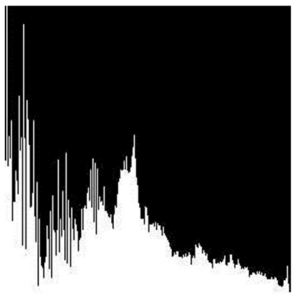
Histogram Stretching (Clipping).

- 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:

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

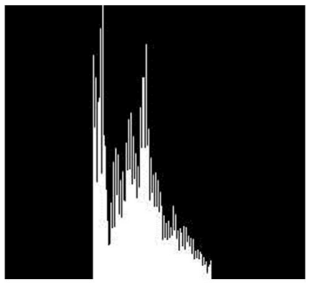
Shrink $_{max}$ and shrink $_{min}$ correspond to the maximum and minimum desired in the compressed histogram. In general, this process produces an image of reduced contrast and may not seem to be useful an image enhancement (see figure below of shrink histogram).

Original image



Histogram of original image

Image after histogram shrink to the range [75, 175]



Histogram of the image

Histogram Shrinking.

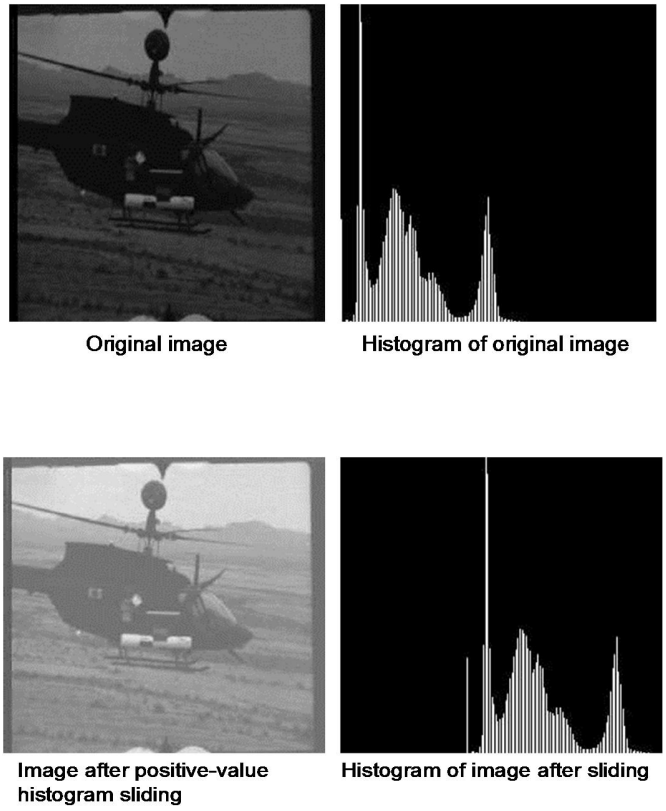
- The histogram slide techniques can be used to make an image either darker or lighter but retain the relationship between gray-level values.

This can be a accomplished by simply adding or subtracting a fixed number for all the gray-level values, as follows:

Slide (I(r,c)) = I (r,c)+ 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.



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 flatting 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 Histogram value	10	8	9	2	14	1	5	2

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:

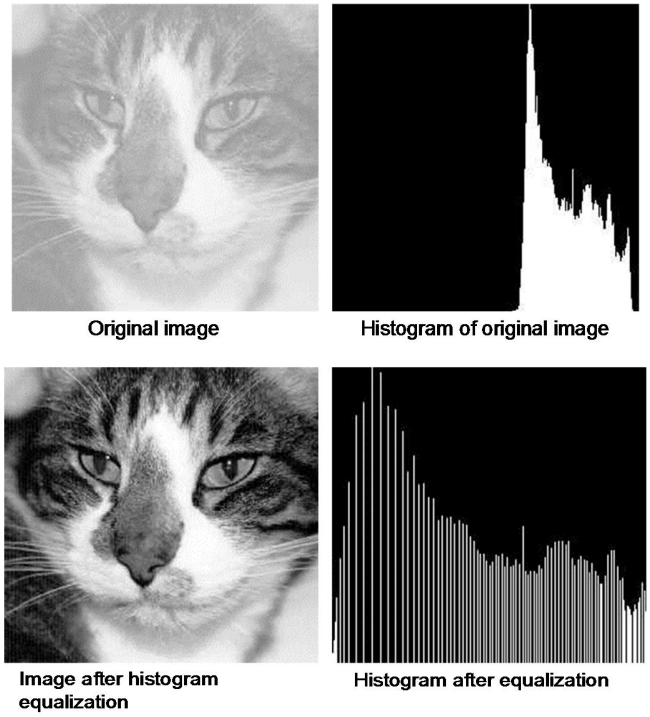
Gray-level	0	1	2	3	4	5	6	7
No. of Pixel	10	8	9	2	14	1	5	2
Run Sum	10	18	27	29	43	44	49	51
Normalized	10/51	18/51	27/51	29/51	43/51	44/51	49/51	51/51
Multiply by 7	1	2	4	4	6	6	7	7

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.

6. Feature Extraction

FEATURE EXTRACTION AND ANALYSIS

The goal in image analysis is to extract information useful for solving **application** based problems. **The first step** to this by reducing the amount of image data with the tools we have explored. **The next step** would be to extract features that are useful in solving computer imaging problems. After we have extracted the features of interest, we can analyze the image

Feature extraction is part of the data reduction process and is followed by feature analysis. One of the important aspects of feature analysis is to determine exactly which features are important, so the analysis is not complete until we incorporate application-specific feedback into the system.

Feature Vectors and Feature Spaces

A feature vector is one method to represent an image, or part of an image (an object), by finding measurements on a set of features. A feature vector is an *n*-dimensional vector that contains a set of values where each value represents a certain feature. This vector can be used to classify an object, or provide us with condensed higher-level information regarding the image.

Let us consider one example:

We need to control a robotic gripper that picks parts from an assembly line and puts them into boxes (either box A or box B, depending on object type). In order to do this, we need to determine:

1) Where the object is 2) What type of object it is

The first step would be to define the feature vector that will solve this problem.

To determine where the object is:

Use the area and the center area of the object, defined by (r,c).

To determine the type of object:

Use the perimeter of object.

Therefore, the feature vector is: [area, r, c, and perimeter]

Distance & Similarity Measures

In feature extraction process, we might need to compare two feature vectors. The primary methods to do this are either to measure the **difference** between the two or to measure the **similarity.** The difference can be measured using a ***distance measure*** in the n-dimensional space. The bigger the distance between two vectors, the greater the difference.

a. Distance Measures

There are several metric measurements: ***Euclidean distance, Range-normalized Euclidean distance, City block*** **or** ***absolute value metric, maximum value***

- ***Euclidean distance*** is the most common metric for measuring the distance between two vectors.

  Given two vectors **A** and **B**, where:

$$A = [a_1 \quad a_2 \quad \ldots \quad a_n]$$
$$B = [b_1 \quad b_2 \quad \ldots \quad b_n]$$

The Euclidean distance is given by:

$$\sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \ldots + (a_n - b_n)^2}$$

This measure may be biased as a result of the varying range on different components of the vector. For example, one component may only range from 1 to 5 and another may range from 1 to 5000, so a difference of 5 for the first component will be maximum, but a difference of 5 for the second feature may be insignificant.

- We can alleviate this problem by using the **range-normalized Euclidean distance,**

$$\sqrt{\sum_{i=1}^{n}\frac{(a_i - b_i)^2}{R_i^2}}$$

R_i is the range of the ith component.

- Another distance measure, called the ***city block* or *absolute value metric***, is defined as follows:

$$\sum_{i=1}^{n}|a_i - b_i|$$

This metric is computationally **faster than the Euclidean distance** but **gives similar result.**

- The final distance metric considered here is the ***maximum value*** metric defined by:

$$\max\{|a_1 - b_1|, |a_2 - b_2|, ..., |a_n - b_n|\}$$

b. Similarity Measures

The second type of metric used for comparing two feature vectors is the ***similarity measure***. The most common form of the similarity measure is the vector **inner product**. Using our definition of vector **A** and **B**, the vector inner product can be defined by the following equation:

$$\sum_{i=1}^{n} a_i b_i = (a_1 b_1 + a_2 b_2 + ... + a_n b_n)$$

When selecting a feature for use in a computer imaging application, an important factor is the **robustness** of the feature. A feature is robust if it will provide consistent results across the entire application domain. For example, if we develop a system to work under any lightning conditions, we do not want to use features that are lightning dependent.

Binary Object Features

In order to extract object features, we need an image that has undergone image segmentation and any necessary morphological filtering. This will provide us with a clearly defined object which can be labeled and processed independently.

After all the binary objects in the image are labeled, we can treat each object as a binary image. The labeled object has a value of '1' and everything else is '0'.

The labeling process goes as follows:

- Define the desired connectivity.
- Scan the image and label connected objects with the same symbol.

After we have labeled the objects, we have an image filled with object numbers. This image is used to extract the features of interest. Among the binary object features include

1. area,
2. center of area,
3. axis of least second moment,
4. perimeter,
5. Euler number,
6. projections,
7. thinness ration and
8. Aspect ratio.

The first four tell us something about **where the object is**, and the latter four tell us something about **the shape of the object**.

In order to provide general equations for area, center of area, and axis of least second moment, we define a function, I,(r, c):

$$I_i(r, c) = \begin{cases} 1 & \text{if } I(r, c) = i\text{th object number} \\ 0 & \text{otherwise} \end{cases}$$

Multiple labeling can occur during sequential scanning, as shown above on the J shaped object. We label two different objects until we reach the pixel

marked X, where we discover that objects 1 and 2 are connected. As show in the figure below

1. **Area**

   The area of the *i*th object is defined as follows:

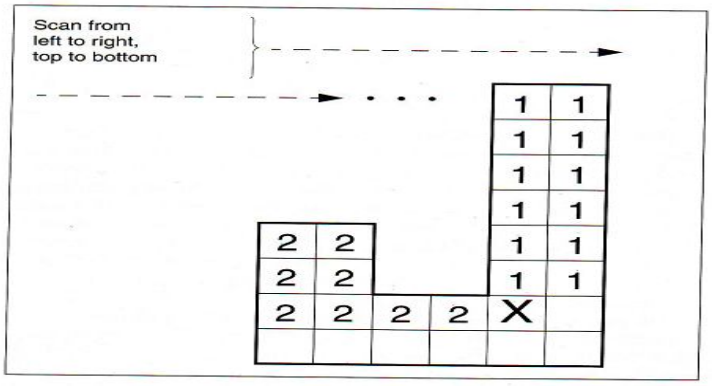
$$A_i = \sum_{r=0}^{height-1} \sum_{c=0}^{width-1} I_i(r,c)$$

   The area A_i is measured in pixels and indicates the relative size of the object.

2. **Center of Area**

   The center of area is defined as follows:

$$\overline{r_i} = \frac{1}{\quad} \sum_{\quad}^{height-1} \sum_{\quad}^{width-1} rI_i(r,c)$$



   These c e center of the *i*th o

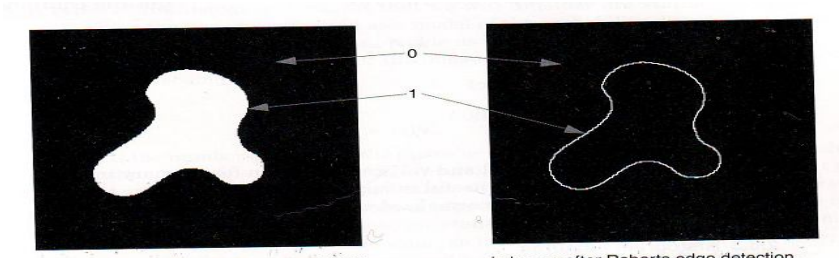
3. **Axis of Least Second Moment**

   The Axis of Least Second Moment is expressed as θ - the angle of the axis relatives to the vertical axis.

4. **Perimeter**

   The perimeter is defined as the total pixels that constitute the edge of the object. Perimeters can be found by counting the number of '1' pixels that have '0' pixels as neighbors. Perimeter can also be found

by applying an edge detector to the object, followed by counting the '1' pixels.



a. Image with binary object. We can find the perimeter by counting the '1' pixels that have a '0' neighbor.

b. Image after Roberts edge detection. We find the perimeter by counting the '1' pixels.

5. Thinness Ratio

The thinness ratio, *T*, can be calculated from perimeter and area.

6. Aspect Ratio

This can be found by scanning the image and finding the minimum and maximum values on the row and column where the object lies.

The equation for aspect ratio is as follows:

$$\frac{c_{\max} - c_{\min} + 1}{r_{\max} - r_{\min} + 1}$$

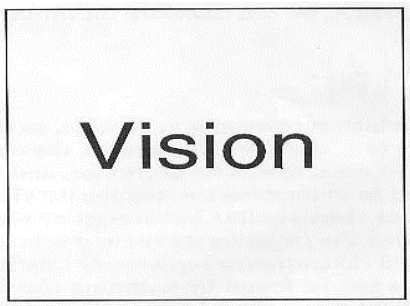
High aspect ratio indicates the object spread more towards **horizontal direction.**

7. Euler Number

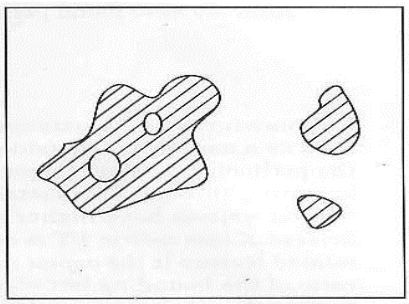
Euler number is defined as the difference between the number of objects and the number of holes.

Euler number = num of object – number of holes

In the case of a single object, the Euler number indicates how many closed curves (holes) the object contains. Euler number can be used in tasks such as optical character recognition (OCR).



a. This image has eight objects and one hole, so its Euler number is 8 − 1 = 7. The letter *V* has Euler number of 1, $i = 2$, $s = 1$, $o = 0$, and $n = 1$.

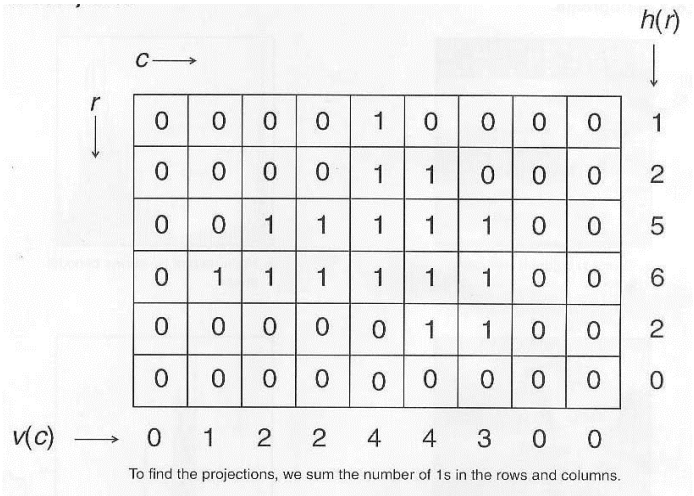


b. This image has three objects and two holes, so the Euler number is 3 − 2 = 1.

8. Projection

The projection of a binary object, may provide useful information related to object's shape. It can be found by summing all the pixels along the rows or columns.

- Summing the rows give horizontal projection.
- Summing the columns give the vertical projection.



To find the projections, we sum the number of 1s in the rows and columns.

7. Feature Extraction

Histogram features

The histogram features that we are considered are statically based features where the histogram is used as a model of the probability distribution of the gray levels. These statistical features provide us with information about the characteristic of the gray – level distribution for the image or sub image. We define the **first–order histogram probability P(a)** as :

$$P(g) = \frac{N(g)}{M}$$

M is the number of pixels in the image or sub image (if the entire image is under consideration, then $M= N^2$ for N×N), and N (g) is the number of pixels at gray level g. As with any probability distribution, all values for P (g) are less than or equal to 1. Histogram features are mean, standard deviation, skew, energy and entropy.

1. **Mean:** the mean is the average value, so it tells us something about the general brightness of the image. A bright image will have a high mean, and a dark image will have a low mean. We will use L as the total number of gray levels available, so the gray levels range from 0 to L-1. For example, for typical 8-bit image data, L is 256 and ranges from 0 to 255. We can define the mean as follows:

$$\bar{g} = \sum_{g=0}^{L-1} gP(g) = \sum_{r}\sum_{c} \frac{I(r,c)}{M}$$

If we use the second form of the equation, we sum over the rows and columns corresponding to the pixels in the image or sub image under consideration.

2. **Standard deviation:** Which is also known as the square root of the **variance**, tell us something about the contrast. It describes the spread in the data, so a high contrast image will have a high variance, and a low – contrast image will have a low variance. It is defined as follows:

$$\sigma = \sqrt{\sum_{g=0}^{L-1} (g - \bar{g})^2 P(g)}$$

3. **Skew** :the skew measure the asymmetry about the mean in the gray-level distribution .it is defined as:

$$\text{SKEW} = \frac{1}{\sigma_g^3} \sum_{g=0}^{L-1} (g - \bar{g})^3 P(g)$$

This method of measuring skew is more computationally efficient, especially considering that, typically, the mean and standard deviation have already been calculated.

4. The **energy** measure tell us something about how the gray level are distributed

$$\text{Energy} = \sum_{g=0}^{L-1} [P(g)]^2$$

The energy measure has a maximum value of 1 for an image with a constant value and gets increasingly smaller as the pixel values are distributed across more gray level values(remember that all the P(g) values are less than or equal to 1). The lager this value is, the easier it is to compress the image data. If the energy is high, it tells us that the number of gray levels in the image is few, that is, the distribution is concentrated in only a small number of different gray levels.

5. **Entropy:** the entropy is a measure that tells us how many bits we need to code the image data and given by :

$$Entropy = -\sum_{g=0}^{L-1} P(g)\log_2[P(g)]$$

As the pixel values in the image are distributed among more gray levels, the entropy increases. This measure tends to vary inversely with the energy. Note (log2=log10 × 3.33).

8. Segmentation

Image segmentation is important in many computer visions and image processing application .The goal of image segmentation is to find regions that represent objects or meaningful parts of objects.

Division of the image into regions corresponding to objects of interest is necessary before any processing can be done at a level higher than that the pixel. Identifying real objects, pseudo—objects, and shadows or actually finding anything of interest within the image requires some form of segmentation.

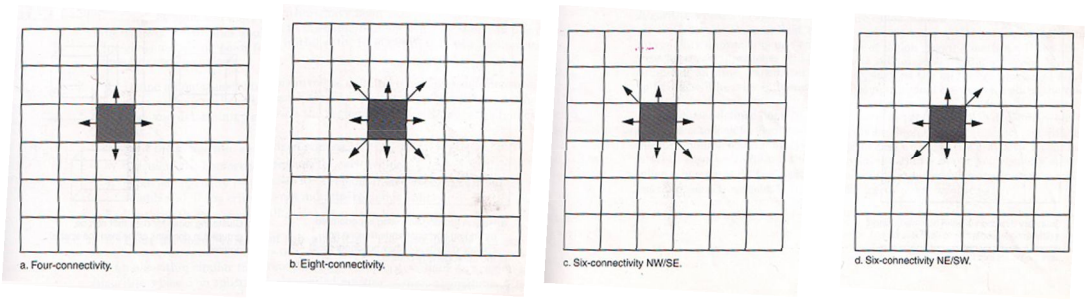
Image Segmentation Methods

Image segmentation methods will look for objects that either have some measure of homogeneity within themselves or have some measure of contrast with the objects on their border. Most image segmentation algorithms are modifications, extensions، or combinations of these two basic concepts. The homogeneity and contrast measures can include features such

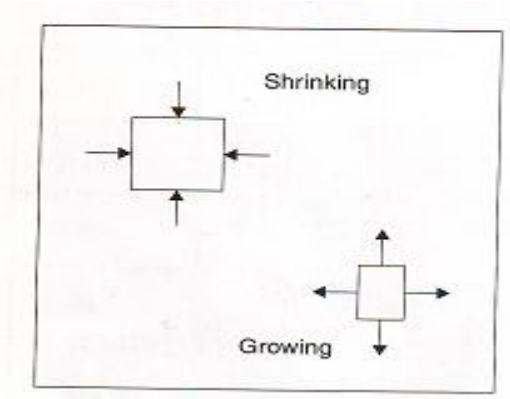
as gray level, color, and texture. After we have performed some preliminary segmentation, we may incorporate higher-level object properties, such as perimeter and shape, into the segmentation process.

Before we look at the different segmentation methods, we need to consider some of the problems associated with image segmentation. The major problems are a result of noise in the image and digitization of a continuous image. Noise is typically caused by the camera, the lenses, the lighting, or the signal path and can be reduced by the use of the preprocessing methods previously discussed. Spatial digitization can cause problems regarding connectivity of objects. These problems can be resolved with careful connectivity definitions and heuristics applicable to the specific domain.

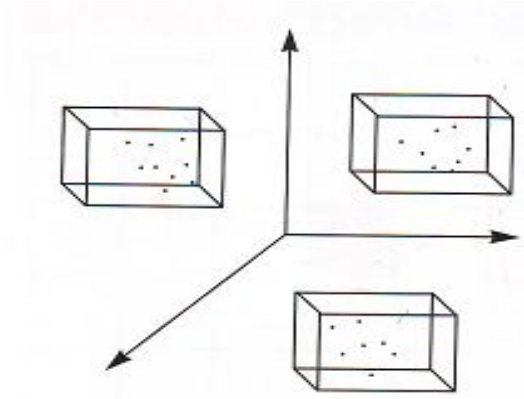
Connectivity refers to the way in which we define an object. After we have segmented an image, which segments should be connected to form an object? Or, at a lower level, when searching the image for homogeneous regions, how do we define which pixels are connected? We must define which of the surrounding pixels are considered to be neighboring pixels. A pixel has eight possible neighbors: two horizontal neighbors, two vertical neighbors, and four diagonal neighbors. We can define connectivity in three different ways: 1) four-connectivity, 2) eight-connectivity, and 3) six-connectivity. Figure below illustrates these three definitions.



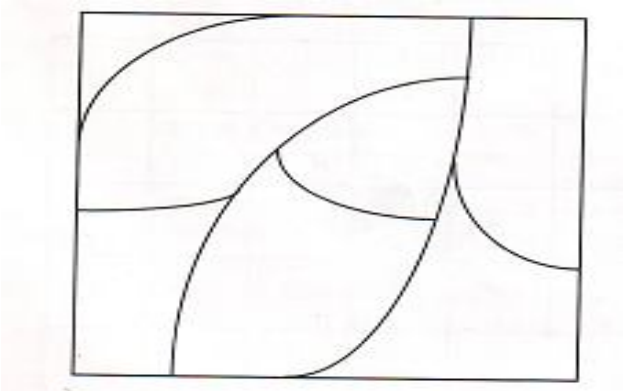
We can divide image segmentation techniques into three main categories (see Figure below): 1) region growing and shrinking, 2) clustering methods, and 3) boundary detection. The region growing and shrinking methods use the row and column (rc) based image space, whereas the clustering techniques can be applied to any domain spatial domain, color space, feature space, etc.



a. Region growing/shrinking is performed by finding homogeneous regions and changing them until they no longer meet the homogeneity criteria.



b. Clustering looks for data that can be grouped in domains other than the spatial domain.



c. Boundary detection is often achieved using a differentiation operator to find lines or edges, followed by postprocessing to connect the points into borders.

1.Region Growing and Shrinking

Region growing and shrinking methods segment the image into regions by operating principally in the RC—based image space. Some of the techniques used are local, in which small areas of the image are processed at a time; others are global, with the entire considered during processing. Methods that can combine local and global techniques, such as split and merge, are referred to as state space techniques and use graph structures to represent the regions and their boundaries.

In general, the split and merge technique proceeds as follows:

1. Define a homogeneity test. This involves defining a homogeneity measure, which may incorporate brightness, color, texture, or other application-specific information, and determining a criterion the region must meet to pass the homogeneity test.
2. Split the image into equally sized regions.
3. Calculate the homogeneity measure for each region.
4. If the homogeneity test is passed for a region, then a merge is attempted with its neighbor(s). If the criterion is not met, the region is split.
5. Continue this process until all regions pass the homogeneity test.

There are many variations of this algorithm. For example, we can start out at the global level, where we consider the entire image as our initial region, and then follow an algorithm similar to the preceding algorithm, but without any region merging. Algorithms based on splitting only are called multi resolution algorithms. Alternately we can start at the smallest level and only merge, with no region splitting. This merge-only approach is one example of region growing methods. Often the results from all these approaches will be quite similar, with the differences apparent only in

computation time. Parameter choice, such as the minimum block size allowed for splitting, will heavily influence the computational burden as well as the resolution available in the results.

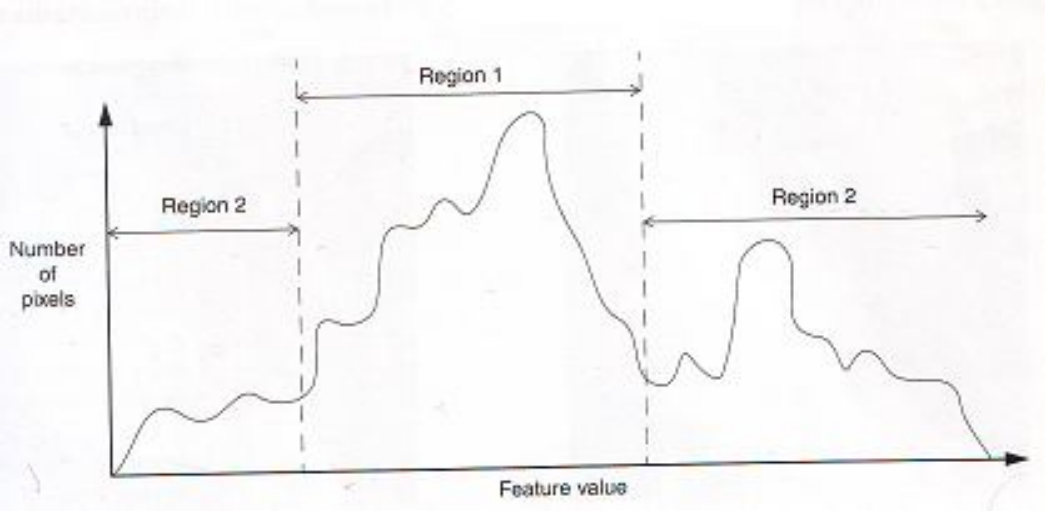
2.Clustering Techniques

Clustering techniques are image segmentation methods by which individual elements are placed into groups; these groups are based on some measure of similarity within the group. The major difference between these techniques and the region growing techniques is that domains other than the rc-based image space (the spatial domain) may be considered as the primary domain for clustering. Some of these other domains include color spaces, histogram spaces, or complex feature spaces. The simplest method is to divide the space of interest into regions by selecting the center or median along each dimension and splitting it there; this can be done iteratively until the space is divided into the specific number of regions needed. This method is used in the SCT Center and PCT/Median segmentation algorithms.

The next level of complexity uses an adaptive and intelligent method to decide where to divide the space. These methods include histogram thresholding and other, more complex feature-space-based statistical methods.

Recursive region splitting is a clustering method that has become a standard technique. This method uses a thresholding of histograms technique to segment the image. A set of histograms is calculated for a specific set of features, and then each of these histograms is searched for distinct peaks.

Histogram peak finding

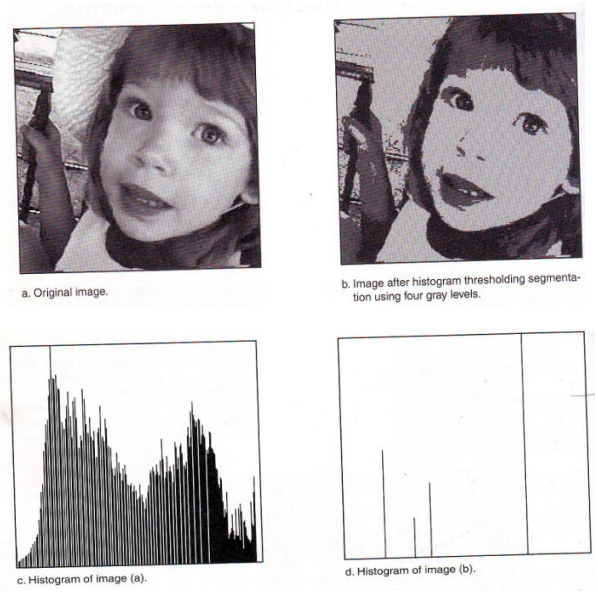


The best peak is selected and the image is split into regions based on this thresholding of the histogram. One of the first algorithms based on these concepts proceeds as follows:

1. Consider the entire image as one region and compute histograms for each component of interest (for example, red, green, and blue for a color image).
2. Apply a peak finding test to each histogram. Select the best peak and put thresholds on either side of the peak. Segment the image into two regions based on this peak.
3. Smooth the binary thresholded image so that only a single connected sub region is left.
4. Repeat steps 1-3 for each region until no new sub regions can be created, that is, no histograms have significant peaks.

Two thresholds are selected, one on each side of the best peak. The image is then split into two regions. Region 1 corresponds to those pixels with feature values between the selected thresh olds, known as those in the peak. Region 2 consists of those pixels with feature values outside the threshold.

Histogram Thresholding Segmentation



a. Original image.

b. Image after histogram thresholding segmentation using four gray levels.

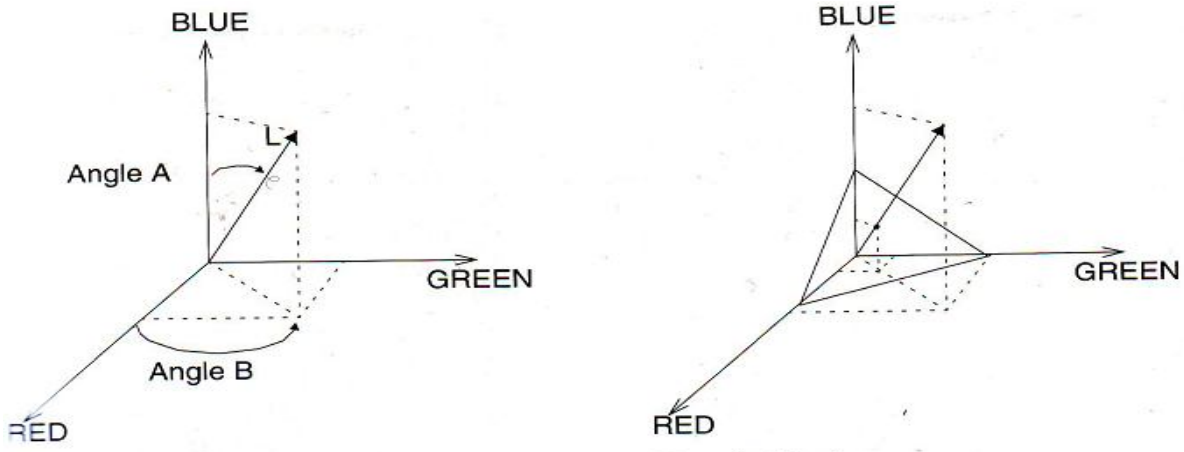
c. Histogram of image (a).

d. Histogram of image (b).

SCT/Center algorithm:

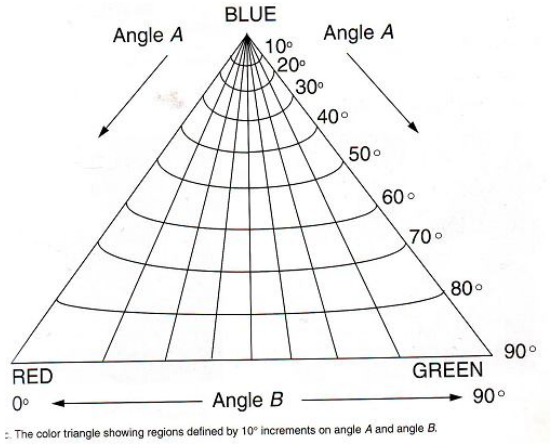
The SCT/Center algorithm (Spherical Coordinate Transform) by using the two- dimensional color subspace defined by two angles we have a more robust algorithm. If we slice a plane through the RGB color space, we can model a color triangle.

SCT/ center and color Triangle



a. The spherical coordinate transform separates the red, green, and blue information into a two-dimensional color space defined by angles *A* and *B*, and a one-dimensional brightness space defined by *L*.

b. The color triangle.

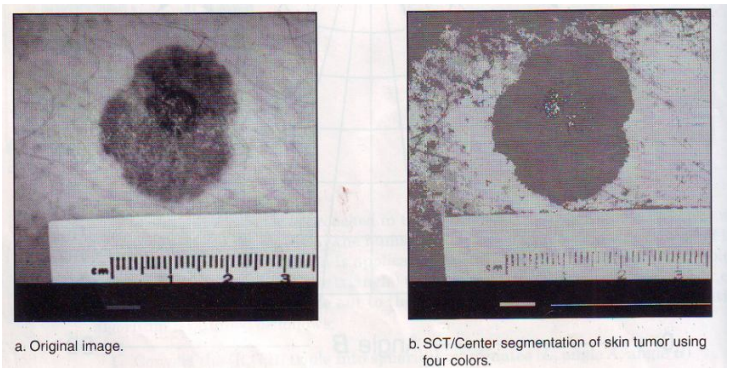


: The color triangle showing regions defined by 10° increments on angle A and angle B.

We can segment the image by taking the color triangle and dividing it into blocks based on limits on the two angles. Figure up shows the shape of the resulting blocks. We can see that for a region defined by a range of minima and maxima on the two angles, the side of the region that is closest to the blue vertex is shorter than the side that is closest to the line that joins the red and green vertices. The SCT/Center segmentation algorithm is outlined as follows:

1. Convert the (R, G, B) triple into spherical coordinates (L, angle A, angle B).
2. Find the minima and maxima of angles A and B.
3. Divide the subspace, defined by the maxima and minima, into equally sized blocks.
4. Calculate the RGB means for the pixel values in each block.
5. Replace the original pixel values with the corresponding RG B means.

For the identification of variegated coloring in the skin tumor application, it was determined that segmenting the image into four colors was optimal. An example of this segmentation method is shown in Figure below.



a. Original image.

b. SCT/Center segmentation of skin tumor using four colors.

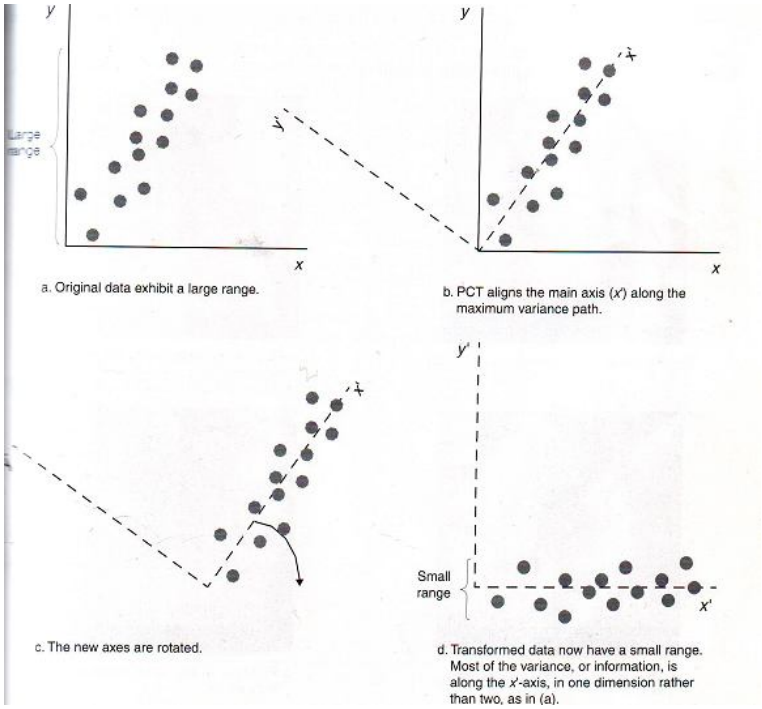
PCT/Median color segmentation algorithm:

This algorithm is based around the principal components transform (PCT). The median split part of the algorithm is based on an algorithm developed for color compression to map 24 bits/pixel color images into images requiring an average of 2 bits/pixel.

The PCT is based on statistical properties of the image and can be applied to any K-dimensional mathematical space. In this case, the PCT is applied to the three-dimensional color space. It was believed that the PCT used in conjunction with the median split algorithm would provide satisfactory color image segmentation because the PCT aligns the main axis along the maximum variance path in the data set.

The PCT/Median segmentation algorithm proceeds as follows:

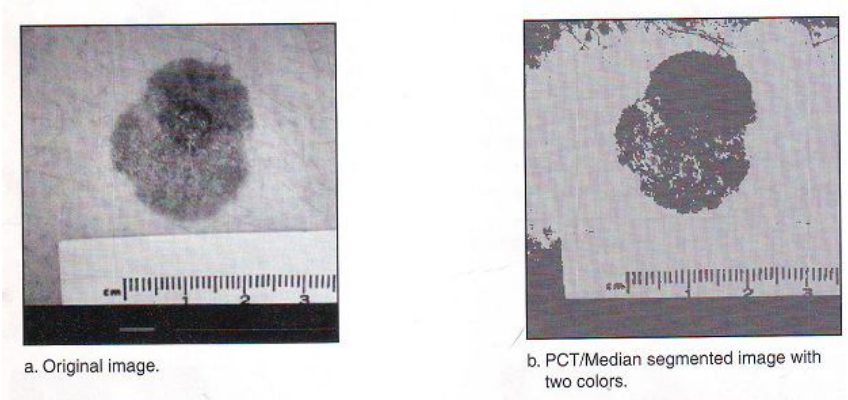
1. Find the PCT for the RGB image. Transform the RGB data using the PCT
2. Perform the median split algorithm: find the axis that has the maximal range (initially it will be the PCT axis). Divide the data along this axis so that there are equal numbers of points on either side of the split the median point. Continue until the desired number of colors is reached.
3. Calculate averages for all the pixels falling within a single box.
4. Map each pixel to the closest average color values, based on a Euclidean distance measure.



Principal Components Transform

Results of this segmentation algorithm are shown in Figure below.

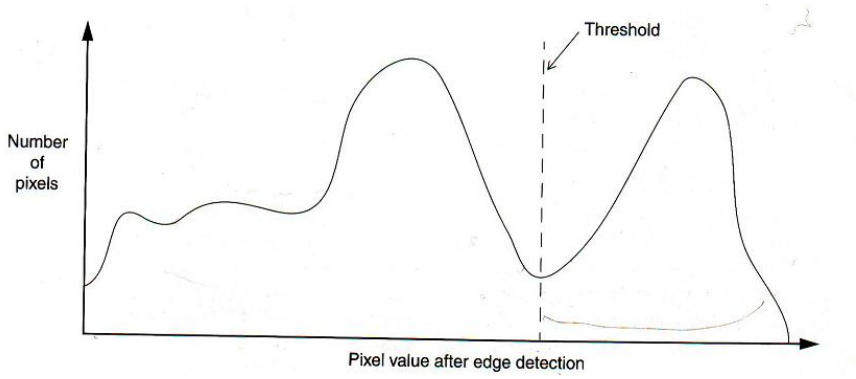
It is interesting to note that the PCT is also used in image compression (coding) since this transform is optimal in the least-square-error sense.



3.Boundary Detection

Boundary detection, as a method of image segmentation, is usually begun by marking points that may be a part of an edge. These points are then merged into line segments, and the line segments are then merged into object boundaries. The edge detectors are used to mark points, thus indicating the possibility of an edge. After the edge detection operation has been performed, the next step is to threshold the results. One method to do this is to consider the histogram of the edge detection results, looking for the best valley.

Edge Detection Threshold



Often, the histogram of an image that has been operated on by an edge detector is unimodal (one peak), so it may be difficult to find a good valley. This method works best with a bimodal histogram. After we have determined a threshold for the edge detection, we need to merge the existing edge segments into boundaries. This is done by edge linking.

9. Segmentation

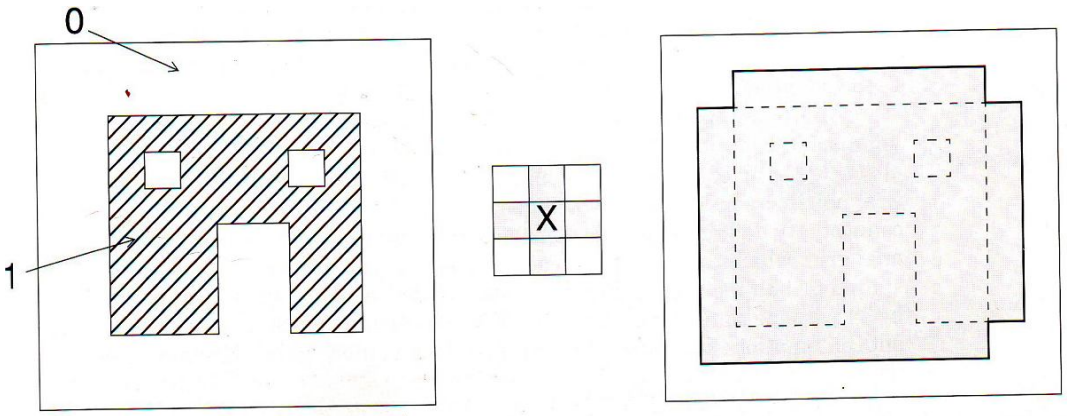
Morphological Filtering

Morphology relates to the structure or form of objects. Morphological filtering simplifies a segmented image to facilitate the search for objects of interest. This is done by smoothing out object outlines, filling small holes, eliminating small projections, and using other similar techniques. Even though this section will focus on applications to binary images, the extension of the concepts to gray-level images will also be discussed. We will look at the different types of operations available and at some examples of their use. The two principal morphological operations are dilation and erosion. Dilation allows objects to expand, thus potentially filling in small holes and connecting disjoint objects. Erosion shrinks objects by etching away (eroding) their boundaries. These operations can be customized for an application by the proper selection of the structuring element, which determines exactly how the objects will be dilated or eroded. The dilation process is performed by laying the structuring element on the image and sliding it across the image in a manner similar to convolution. The difference is in the operation performed. It is best described in a sequence of steps:

1. If the origin of the structuring element coincides with a '0' in the image, there is no change; move to the next pixel.
2. If the origin of the structuring element coincides with a '1' in the image, perform the OR logic operation on all pixels within the structuring element.

An example is shown in Figure below.

Dilation



a. Original image. b. Structural element; x = origin. c. Image after dilation; original in dashes.

Note that with a dilation operation, all the '1' pixels in the original image will be retained, any boundaries will be expanded, and small holes will be filled.

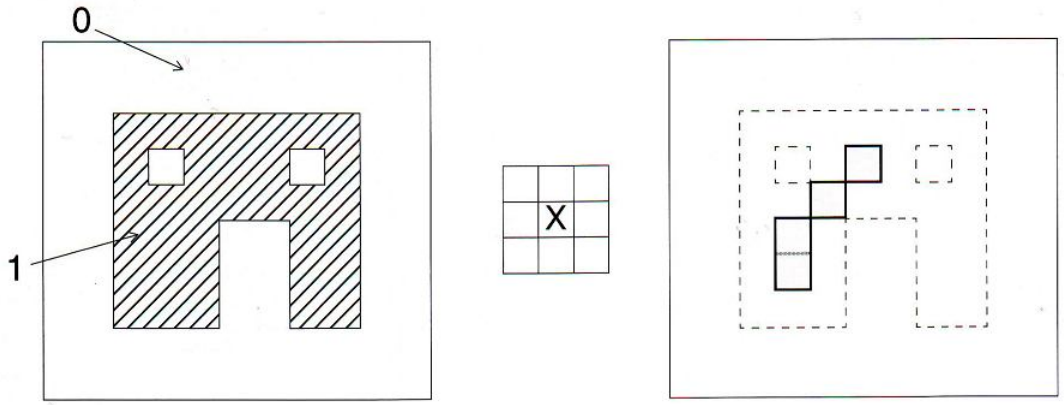
The erosion process is similar to dilation, but we turn pixels to 'O', not '1'. As before, slide the structuring element across the image and then follow these steps:

1. If the origin of the structuring element coincides with a '0' in the image, there is no change; move to the next pixel.

2. If the origin of the structuring element coincides with a '1' in the image, and any of the '1' pixels in the structuring element extend beyond the object ('1' pixels) in the image, then change the '1' pixel in the image to a '0'.

In Figure below, the only remaining pixels are those that coincide to the origin of the structuring element where the entire structuring element was contained in the existing object.

Erosion

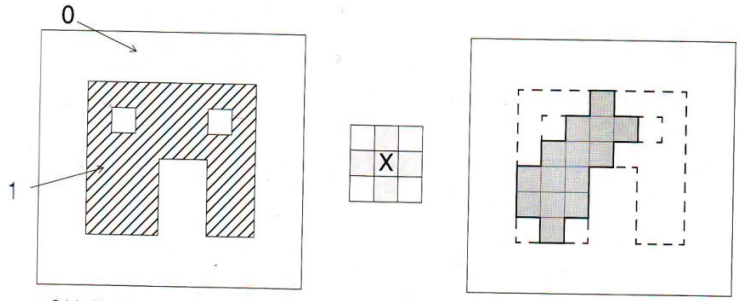


a. Original image. b. Structural element; x = origin c. Image after erosion; original in dashes.

Because the structuring element is 3 pixels wide, the 2-pixel-wide right leg of the image object was eroded away, but the 3-pixel-wide left leg retained some of its center pixels.

These two basic operations, dilation and erosion, can be combined into more complex sequences. The most useful of these for morphological filtering are called opening and closing. Opening consists of an erosion followed by a dilation and can be used to eliminate all pixels in regions that are too small to contain the structuring element. See Figure below for an example of opening.

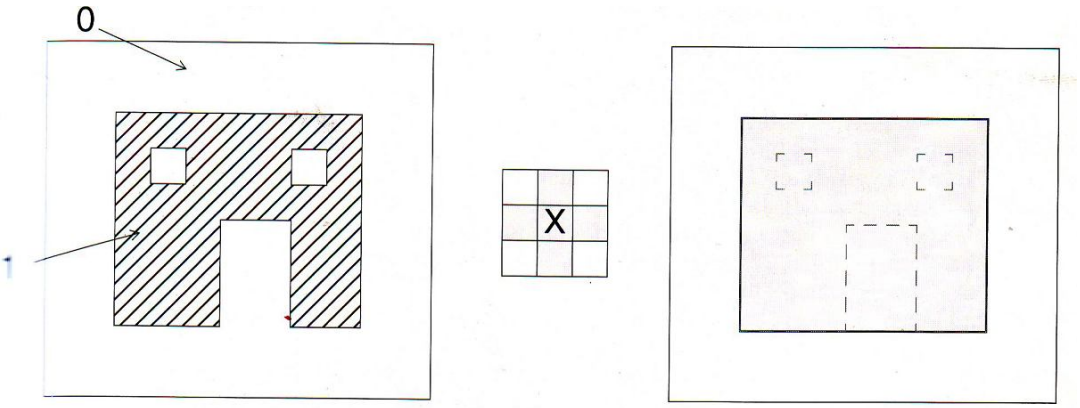
Opening



a. Original image. b. Structural element; x = origin. c. Image after opening; erosion followed by dilation.

Closing consists of a dilation followed by erosion and can be used to fill in holes and small gaps. In Figure below we see that the closing operation has the effect of filling in holes and closing gaps.

Closing



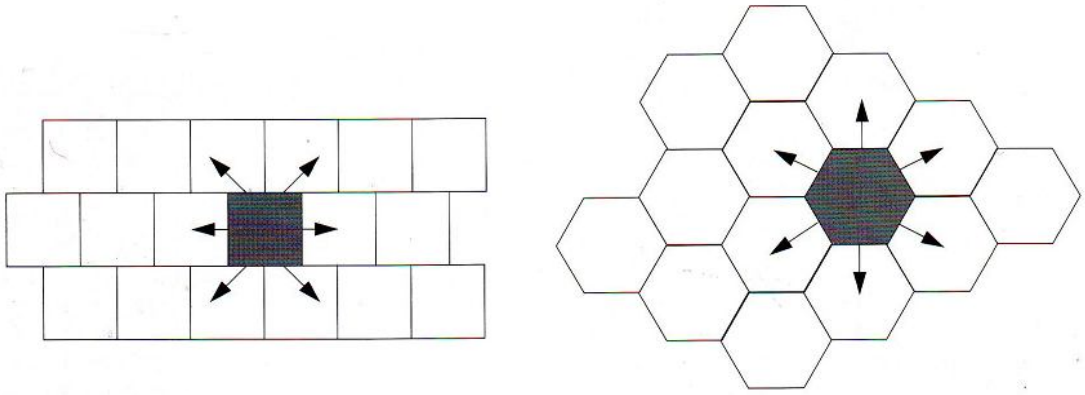
a. Original image. b. Structural element; x = origin. c. Image after closing; dilation followed by erosion; original in dashes.

Comparing Figure closing to Figure opening, we see that the order of operation is important. Closing and opening will have different results even though both consist of an erosion and a dilation.

Another approach to binary morphological filtering is based on an iterative approach. The usefulness of this approach lies in its flexibility. It is based on a definition of six-connectivity; in which each pixel is considered connected to its horizontal and vertical neighbors but to only two diagonal neighbors (the two on the same diagonal). This connectivity definition is equivalent to assuming that the pixels are laid out on a hexagonal grid, which can be

simulated on a rectangular grid by assuming that each row is shifted by half a pixel (see Figure below).

Hexagonal Grid

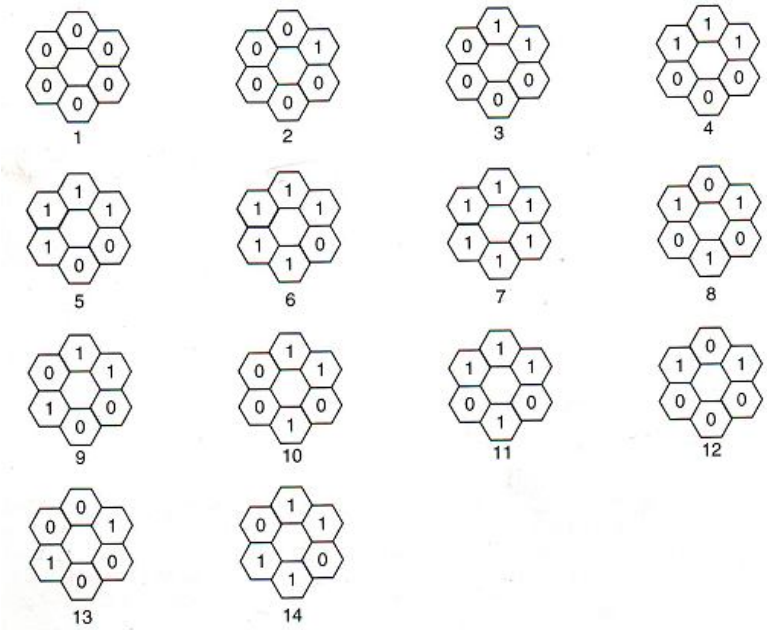


a. Rectangular image grid with every other row shifted by one-half pixel.

b. Hexagonal grid.

With this definition a pixel can be surrounded by 14 possible combinations of 1's and 0's, as seen in Figure below;

Surrounds for iterative Morphological Filtering



we call these different combinations surrounds. For this approach to morphological filtering, we define:

1. The set of surrounds S, where a = 1.
2. A logic function, L(a, b), where b is the current pixel value, specifies the output of the morphological function.

The function L( ) and the values of a and b are all functions of the row and column, (r, c), but for concise notation this is implied. Set S can contain any or all of the14 surrounds defined in Figure up. L(a, b) can be any logic function, but it turns out that the most useful are the AND and OR functions. The AND function tends to etch away at object boundaries (erosion), and the OR function tends to grow objects (dilation).

EXAMPLE :

Let S = {2, 3, 4, 5, 6} and L = a + b (+ = OR). Because L(a, b) is an OR operation, all pixels that are 1 in the original will remain 1. The only pixels that will change are those that are 0 in the original image and have a surround that is S (this means that a = 1). If we examine the set S, we see that this set contains all pixels that are surrounded by a connected set of 1's. This operation will expand the object, but because the surrounds of disconnected 1 pixels are not included in S, disjoint objects will not connect. We can see from this example that this method is more flexible than the methods described earlier. We can use this technique to define methods for dilation, erosion, opening, and closing, as well as others. For this technique the selection of the set S is comparable to defining the structuring element in the previously described approaches, and the operation L(a, b) defines the type of filtering that occurs.

10. MV Techniques

This lecture introduces "low-level" vision processing operators, which lack the ability to perform the perfect functions of intelligence: logical deduction, searching, classification and learning. Many intermediate-level operations can be implemented by concatenating these basic commands in simple macros.

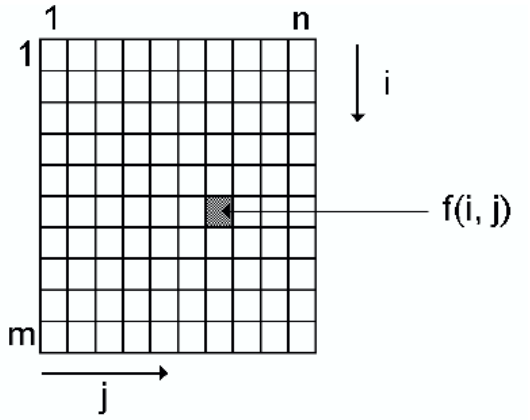
Representations of Images

We shall first consider the representation of *Monochrome* (grey-scale) images. Let i and j denote two integers where $1 \leq i \leq m$ and $1 \leq j \leq n$. In addition, let $f(i,j)$ denote an integer function such that $0 \leq f(i,j) \leq W$. (W denotes the white level in a grey-scale image.) We may *arbitrarily* assign intensities according to the following scheme:

$f(i,j) = 0$	black
$0 < f(i,j) \leq 0.33W$	dark grey
$0.33W < f(i,j) \leq 0.67W$	mid-grey
$0.67W < f(i,j) < W$	light grey
$f(i,j) = W$	white

Elementary Image Processing Functions

The following notation will be used throughout this section, in which we shall concentrate upon grey-scale images, unless otherwise stated:



- i and j are row and column address variables and lie within the ranges: $1 \leq i \leq m$ and $1 \leq j \leq$ n.
- $A = \{a(i,j)\}$, $B = \{b(i,j)\}$ and $C = \{c(i,j)\}$.
- W denotes the white level.
- $g(X)$ is a function of a single independent variable X.
- $h(X,Y)$ is a function of two independent variables, X and Y.
- The assignment operator '←' will be used to define an operation that is performed upon one data element. In order to indicate that an operation is to be performed upon all pixels within an image, the assignment operator '⇐' will be used.
- k, k1, k2, k3 are constants.
- $N(i,j)$ is that set of pixels arranged around the pixel (i,j) in the following way:

$(i\text{-}1, j\text{-}1)$	$(i\text{-}1, j)$	$(i\text{-}1, j\text{+}1)$
$(i, j\text{-}1)$	(i, j)	$(i, j\text{+}1)$
$(i\text{+}1, j\text{-}1)$	$(i\text{+}1, j)$	$(i\text{+}1, j\text{+}1)$

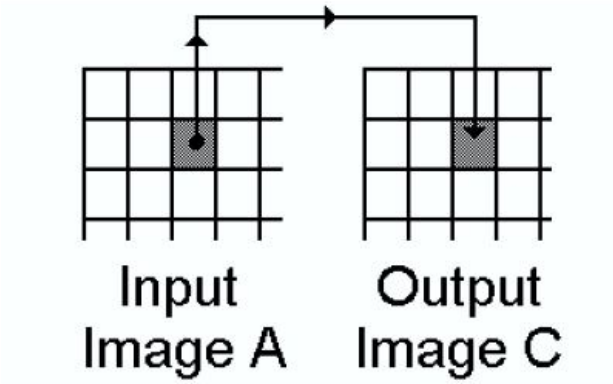
Notice that $N(i,j)$ forms a 3×3 set of pixels and is referred to as the 3×3 *neighbourhood* of (i,j).

Monadic, Point-by-point Operators.

These operators have a characteristic equation of the form:

$$c(i,j) \Leftarrow g(a(i,j)) \text{ or } E \Leftarrow g(E)$$

Such an operation is performed for all (i,j) in the range $[1,m].[1,n]$ as figure below:



Monadic point-by-point operator. The (i,j)th pixel in the input image has intensity $a(i,j)$. This value is used to calculate $c(i,j)$, the intensity of the corresponding pixel in the output image.

Several examples will now be described.

1. ***Intensity shift***

$$c(i,j) \Leftarrow \begin{cases} 0 & a(i,j)+k<0 \\ a(i,j)+k & 0 \le a(i,j)+k \le W \\ W & W < a(i,j)+k \end{cases}$$

k is a constant, set by the system user. Notice that this definition was carefully designed to maintain $c(i,j)$ within the same range as the input $[0,W]$. This is an example of a process referred to as *intensity normalisation*.

2. *Intensity multiply*

$$c(i,j) \Leftarrow \begin{cases} 0 & a(i,j) \,.\, k < 0 \\ a(i,j) \,.\, k & 0 \le \mathrm{a(i,j)} \,.\, \mathrm{k} \le \mathrm{W} \\ W & W < a(i,j) \,.\, k \end{cases}$$

3. *Logarithm*

$$c(i,j) \Leftarrow \begin{cases} 0 & a(i,j) = 0 \\ W.\log(a(i,j))/\log(W) & \text{otherwise} \end{cases}$$

This definition arbitrarily replaces the infinite value of log(0) by zero, and thereby avoids a difficult rescaling problem.

4. *Antilogarithm (exponential)*

$$c(i,j) \Leftarrow W.\ \exp(a(i,j))/\exp(W)$$

5. *Negate*

$$c(i,j) \Leftarrow W - a(i,j)$$

6. *Threshold*

$$c(i,j) \Leftarrow \begin{cases} W & k1 \le a(i,j) \le k2 \\ 0 & \text{otherwise} \end{cases}$$

This is an important function, which converts a grey-scale image to a binary format. Unfortunately, it is often difficult to find satisfactory values for the parameters *k1* and *k2*.

7. Highlight

$$c(i,j) \Leftarrow \begin{cases} k3 & k1 \le a(i,j) \le k2 \\ a(i,j) & \text{otherwise} \end{cases}$$

8. Squaring

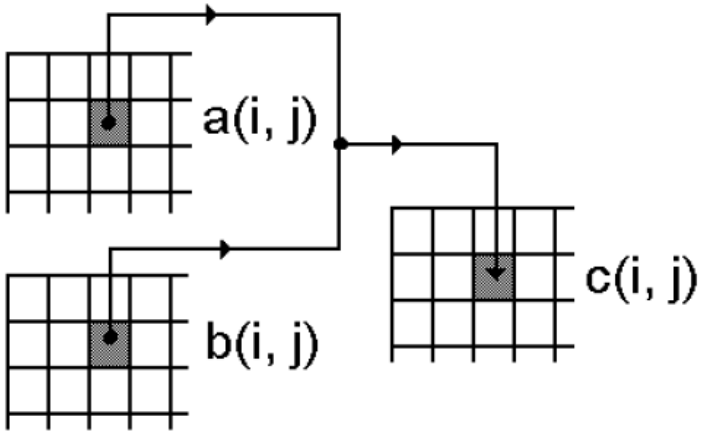
$$c(i,j) \Leftarrow [a(i,j)]^2/W$$

Dyadic Point-by-point Operators

Dyadic operators have a characteristic equation of the form:

$$c(i,j) \Leftarrow h(a(i,j), b(i,j))$$

There are two input images: $A = \{a(i,j)\}$ and $B = \{b(i,j)\}$, while the output image is $C = \{c(i,j)\}$. It is important to realise that $c(i,j)$ depends upon only $a(i,j)$ and $b(i,j)$.



Dyadic point-by-point operator. The intensities of the (i,j)th pixels in the two input images (i.e., $a(i,j)$ and $b(i,j)$) are combined to calculate the intensity, $c(i,j)$, at the corresponding address in the output image.

Here are some examples of dyadic operators:

Add [add]

$$c(i,j) \Leftarrow [a(i,j) + b(i,j)]/2.$$

Subtract [sub]

$$c(i,j) \Leftarrow [(a(i,j) - b(i,j)) + W]/2$$

Multiply [mul]

$$c(i,j) \Leftarrow [a(i,j).b(i,j)]/W$$

Maximum [max]

$$c(i,j) \Leftarrow MAX\,[a(i,j), b(i,j)]$$

When the maximum operator is applied to a pair of binary images, the *union* (OR function) of their white areas is computed.

Minimum [min]

$$c(i,j) \Leftarrow MIN\,[a(i,j), b(i,j)]$$

When A and B are both binary, the *intersection* (AND function) of their white areas is calculated.

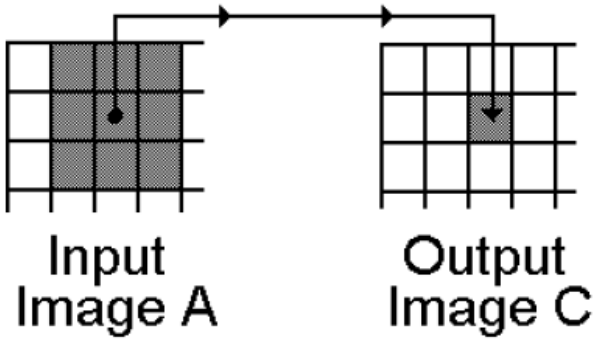
Local Operators

The figure below illustrates the principle of the operation of local operators. Notice that the intensities of several pixels are combined together, in order to calculate the intensity of just one pixel. Amongst the simplest of the local operators are those which use a set of nine pixels arranged in a 3 × 3 square. These have a characteristic equation of the following form:

$$c(i,j) \Leftarrow g(a(i\text{-}1, j\text{-}1), a(i\text{-}1, j), a(i\text{-}1, j\text{+}1), a(i, j\text{-}1), a(i, j), a(i, j\text{+}1), a(i\text{+}1, j\text{-}1), a(i\text{+}1, j), a(i\text{+}1, j\text{+}1))$$

where $g(.)$ is a function of 9 variables. This is an example of a local operator, which uses a *3 × 3 processing window*. In the simplified notation which we introduced earlier, the above definition reduces to:

$$E \Leftarrow g(A, B, C, D, E, F, G, H, I)$$



Local operator. In this instance, the intensities of nine pixels arranged in a 3 × 3 window are combined together. Local operators may be defined which uses other, possibly larger windows. The window may, or may not, be square and the calculation may involve linear or non-linear processes.

Linear Local Operators

An important sub-set of the local operators is that group, which performs a linear, weighted sum, and which are therefore known as *linear local operators*. For this group, the characteristic equation is:

$$E \Leftarrow k1.(A.W1 + B.W2 + C.W3 + D.W4 + E.W5 + F.W6 + G.W7 + H.W8 + I.W9) + k2$$

where $W1$, $W2$,...,$W9$ are weights, which may be positive, negative or zero. Values for the normalisation constants, $k1$ and $k2$ are given later. The matrix

illustrated below is termed the *weight matrix* and is important, because it determines the properties of the linear local operator.

$W1$	$W2$	$W3$
$W4$	$W5$	$W6$
$W7$	$W8$	$W9$

The following rules summarise the behaviour of this type of operator:

1. If all weights are either positive or zero, the operator will *blur* the input image. Blurring is referred to as *low-pass filtering*. Subtracting a blurred image from the original results in a highlighting of those points where the intensity is changing rapidly and is termed *high-pass filtering*.
2. If $W1 = W2 = W3 = W7 = W8 = W9 = 0$, and $W4$, $W5$, $W6 > 0$, then the operator blurs along the rows of the image; horizontal features, such as edges and streaks, are not affected.
3. If $W1 = W4 = W7 = W3 = W6 = W9 = 0$, and $W2$, $W5$, $W8 > 0$, then the operator blurs along the columns of the image; vertical features are not affected.
4. If $W2 = W3 = W4 = W6 = W7 = W8 = 0$, and $W1$, $W5$, $W9 > 0$, then the operator blurs along the diagonal (top-left to bottom-right). There is no smearing along the orthogonal diagonal.
5. If the weight matrix can be reduced to a matrix product of the form $P.Q$, where

$P =$

0	0	0
V4	V5	V6
0	0	0

and

$$Q = \begin{array}{|c|c|c|}\hline 0 & V1 & 0 \\ \hline 0 & V2 & 0 \\ \hline 0 & V3 & 0 \\ \hline \end{array}$$

the operator is said to be of the "separable" type.

6. In order to perform normalisation, the following values are used for $k1$ and $k2$.

$$k1 \leftarrow 1/\sum_{p,q}\left|W_{p,q}\right|$$

$$k2 \leftarrow \left[1-\sum_{p,q}W_{p,q}/\sum_{p,q}\left|W_{p,q}\right|\right]\cdot W/2$$

Non-linear Local Operators

1. ***Largest intensity neighbourhood function***

$$E \Leftarrow \mathbf{MAX}(A, B, C, D, E, F, G, H, I)$$

This operator has the effect of spreading bright regions and contracting dark ones.

2. ***Crack detector***

This operator is equivalent to applying the above operation and then subtracting the result from the original image. This detector is able to detect thin dark streaks and small dark spots in a grey-scale image; it ignores other features, such as bright spots and streaks, edges (intensity steps) and broad dark streaks.

3. *Rank filters*

The generalised 3 × 3 rank filter is:

$$c(i,j) \Leftarrow k1.(A'.W1 + B'.W2 + C'.W3 + D'.W4 + E'.W5 + F'.W6 + G'.W7 + H'.W8 + I'.W9) + k2$$

where

A' = **LARGEST** (A, B, C, D, E, F, G, H, I)

B' = **SECOND_LARGEST** (A, B, C, D, E, F, G, H, I)

C' = **THIRD_LARGEST** (A, B, C, D, E, F, G, H, I)

….

I' = **NINTH_LARGEST** (A, B, C, D, E, F, G, H, I)

and $k1$ and $k2$ are the normalisation constants defined previously. With the appropriate choice of weights ($W1$, $W2$, ..., $W9$), the rank filter can be used for arange of operations including edge detection, noise reduction, edge sharping and image enhancement.

4. *Direction codes*

This function can be used to detect the *direction* of the intensity gradient. A direction code function DIR_CODE is defined thus:

$$\text{DIR\_CODE}(A,B,C,D,F,G,H,I) \Leftarrow \begin{cases} 1 & \text{if } A \geq \text{MAX}(B,C,D,F,G,H,I) \\ 2 & \text{if } B \geq \text{MAX}(A,C,D,F,G,H,I) \\ 3 & \text{if } C \geq \text{MAX}(A,B,D,F,G,H,I) \\ 4 & \text{if } D \geq \text{MAX}(A,B,C,F,G,H,I) \\ 5 & \text{if } F \geq \text{MAX}(A,B,C,D,G,H,I) \\ 6 & \text{if } G \geq \text{MAX}(A,B,C,D,F,H,I) \\ 7 & \text{if } H \geq \text{MAX}(A,B,C,D,F,G,I) \\ 8 & \text{if } I \geq \text{MAX}(A,B,C,D,F,G,H) \end{cases}$$

Using this definition the operator *dbn* may be defined as:

$E \Leftarrow \text{DIR\_CODE}(A,B,C,D,F,G,H,I)$

11.Pattern Recognition

1. Basic concepts of pattern recognition

A pattern is the description of an object. We can spot a friend in a crowd and recognize what he says, we can recognize the voice of a known individual, we can read handwriting and analyze fingerprints, we can distinguish smiles from gestures of anger.

We can divide our acts of recognition into two major types: the recognition of ***concrete*** items and the recognition of ***abstract*** items.

- Concrete items : we recognize characters, pictures, music, and object around us. This may be referred to as sensory recognition, which includes visual and aural pattern recognition. This recognition process involves the identification and classification of spatial and temporal patterns.
- Abstract items : we can recognize an old argument, or a solution to a problem, with our eyes and ears closed. This process involves the recognition of abstract items and can be termed ***conceptual recognition.***

We will concentrate on the first type, which include ***spatial patterns*** : characters, fingerprints, weather maps, physical objects, and pictures, and ***temporal patterns*** : speech, speaker, target signature.

Pattern recognition can be defined as the categorization of input data into identifiable classes via the extraction of significant features or attributes of the data from a background of irrelevant detail.

Table (1) describes several classification tasks, together with the corresponding input data and output responses.

Task of Classification	Input Data	output Response
Character recognition	Optical signal	Name of character
Speech recognition	Acoustic waveform	Name of word
Speaker recognition	Voice	Name of speaker
Weather prediction	Weather maps	Weather forecast
Medical diagnosis	Symptoms	Disease

Table (1)

An obvious but simple-minded solution to a pattern recognition problem is to perform a number of simple tests on the individuals input patterns in order to ***extract the features*** of each pattern class. Such tests should be sufficient to distinguish between permissible input pattern that belong to different classes. Consider for instant the following four Chinese characters:

these simple characters may be recognized by performing tests on the existence of a ***vertical stroke, a horizontal stroke, a single dote, an open bottom, an open top, a dot sequence, and by counting the number and sequence of strokes.***

As a second example consider the following five English letters :

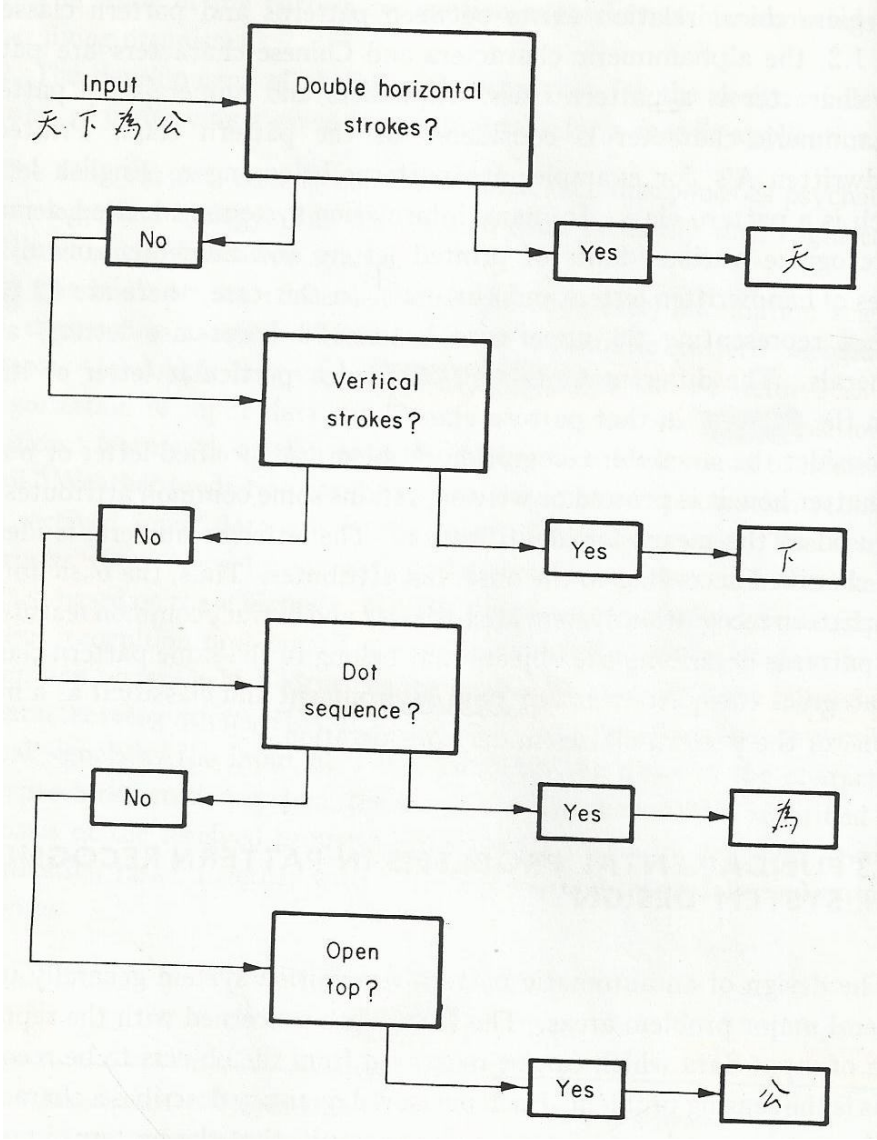
C O I N S

These letters can be classified by making test on the existence of such features as ***a lake, a single bay, a double bay, a vertical line, and a short line.***

A functional block diagram illustrate the pattern recognition concept described above is shown in figure below.

A hierarchical relation exists between patterns and pattern classes. Alphabets and numerals are patterns if alphanumeric character is considered as the pattern class. Printed and handwritten A's for example, are pattern belonging to the English letter A, which is a pattern class. In many information systems, we need a machine to recognize various fonts of printed letters and numerals, and different styles of handwritten letters and numerals. In this case, there are ***62 pattern classes*** representing 26 upper-case letters, 26 lower-case letters, and 10 numerals. the ***different fonts and styles*** of a particular letter or numeral form the ***patterns*** in that pattern class.

Consider the character recognition problem. A specified letter or numeral, no matter how it is printed or written, retains some common attributes which are used as the means for identification. The letter or numeral is identified and classified according to the observed attributes. Thus, the ***basic function*** of a pattern recognition system are to ***detect and extract common features*** from the pattern describing the objects that belonging to the same pattern class, and to ***recognize*** this pattern in ***any new environment*** and classify it as member of one of the pattern classes under consideration.



A simple scheme for classifying characters

2. Fundamental problems in pattern recognition system design

The design of an automatic pattern recognition system generally involves several major problem areas.

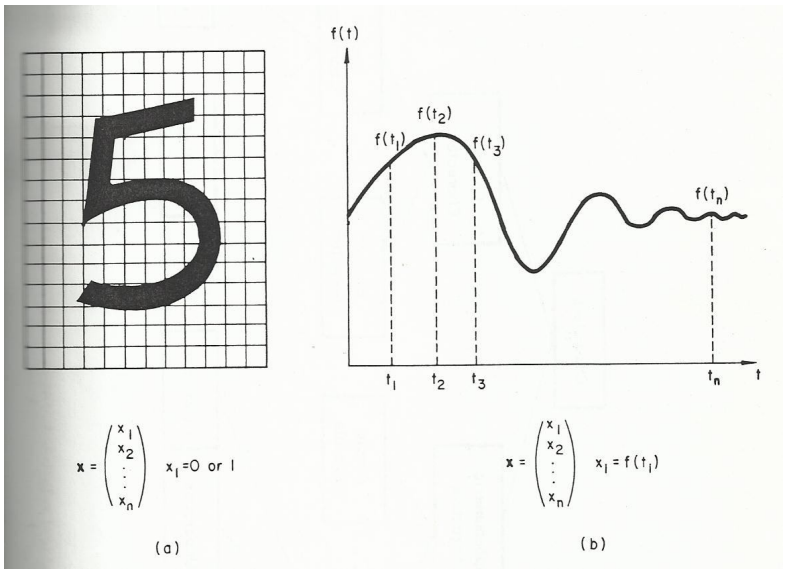
First : is concerned with the representation of the input data which can be measured from the objects to be recognized. This is the sensing problem.

Each measured quantity describes a characteristic of the pattern or object. Suppose, for example, that the pattern in question are alphanumeric characters. In this case, a grid measuring scheme such as the one shown in figure below (a) can be effectively used in the sensor. If we assume that the grid has n elements, the measurements can be arranged in the form of a ***pattern vector*** :

$$X = \begin{pmatrix} x_1 \\ x_2 \\ . \\ x_n \end{pmatrix}$$

where each element x_i is , for example, assigned the value 1 if the ith cell contains a portion of the character, and is assigned the value 0 otherwise.

A second example is shown in figure (b) , in this case, the patterns are continuous functions (such as acoustic signals) of a variable t. If these functions are sampled at discrete points $t_1, t_2, \ldots, t_n$ a pattern vector may be formed by letting $x_1=f(t_1), x_2=f(t_2),\ldots, x_n = f(t_n)$.



Two simple scheme for the generation of pattern vector

The ***pattern vectors*** contain all the measured information available about the patterns. The measurements performed on the objects of a pattern class may be regarded as a coding process which consist of assigning to each pattern characteristic a symbol from the alphabet set$\{x_i\}$. When the measurements yield information in the form of real numbers, it is often useful to think of a pattern vector as a point in the n-dimensional Euclidean space.

Second: this problem concerns the ***extraction*** of character ***features*** or attributes from the received input data and the ***reduction*** of ***dimensionality*** of pattern vectors. This is often referred to as the preprocessing and feature extraction problem.

In speech recognition for example, we may discriminate vowels and vowel-like sounds from fricative and certain other consonants by measuring the distribution of energy over frequency in the spectra. The commonly used ***feature for speech recognition*** are the ***duration of sound, the ratios of energy in various frequency band, the location of spectral peaks, or formats, and the movement of these peaks in time.***

If a complete set of ***discriminatory (مميزه) features*** for each pattern class can be determined from the measured data, the recognition and classification of patterns will present ***little difficulty***.

Correct recognition will depend on :

1. The amount of ***discrimination information*** contained in the measurement .
2. The ***effective utilization*** of this information.

Third : this problem in pattern recognition system design involves the determination of ***optimum decision procedures***, which are needed in the identification and classification process. After the observed data from patterns to be recognized have been expressed in the form of pattern points or measurement vector in the pattern space, ***we want the machine to decide to which pattern class these data belong.***

The decision functions can be generated in a variety of ways :

1. When complete a ***prior knowledge*** about the patterns to be recognized is ***available***, the decision functions may be determined with ***precision*** دقة on the basis of this information.
2. When only ***qualitative knowledge*** about the patterns is ***available***, ***reasonable guesses*** of the forms of the decision functions can be made. In this case the decision boundaries may be ***far from correct***, and it is necessary to design the machine to achieve satisfactory performance through a ***sequence of adjustments***.
3. The more general situation is that there exists ***little***, if any, a ***priori knowledge*** about the patterns to be recognized. Under these circumstances pattern-recognizing machines are best designed using a ***training or learning procedure.***
4. If all possible characteristics can be measured and ***unlimited time*** is available for processing the measured information, a ***brute-force*** technique may be applied to achieve quite adequate pattern recognition. In usual practice, however, ***restriction in time, space, and cost*** dictate the development of realistic approaches.

Characteristic of adaptive pattern recognition

1. Resistance مقاوم to distortions.

2. Flexible under large pattern deviationsانحرافات.
3. Capable of self-adjustment تعديل.

3. Design Concepts and Methodologies

Three basic design concepts are discussed in the following paragraphs.

1. Membership-roster concept

Characterization of pattern class by roster of its members suggest automatic pattern recognition by *template matching*. The set of patterns belonging to the same pattern class is stored in the pattern recognition system. When an unknown pattern is shown to the system, it is compared with the stored patterns one by one. The pattern recognition system classifies this input pattern as a member of a pattern class if it matches one of the stored patterns belonging to that pattern class.

2. Common-property concept

Characterization of pattern class by common properties shared by all of its members suggests automatic pattern recognition via the detection and processing of **similar features**. The basic assumption in this method is that ***patterns belonging to the same class posses certain common properties or attributes which reflect similarities among these patterns.***

Its appears that this concept ***excels*** the membership-roster approach in many concepts:

- The storage requirement for the features of a pattern class is ***much less*** severe than that for all the patterns in the class.
- since features of a pattern class are ***invariant*** , comparison of features allows variation in individual pattern.

- Utilization of this concept necessitates the development of feature selection techniques which are optimum in some sense.

3. **Clustering concept**
   - When the patterns of a class are vectors whose components are ***real numbers***, a pattern class can be characterized by its clustering properties in the pattern space. The design of pattern recognition system based on this general concept is guided by the relative ***geometrical arrangement*** of the various pattern clusters.
   - If the class are characterized by clusters which are ***far apart***, simple recognition schemes such as the ***minimum-distance classifiers*** may be successfully employed.
   - When the ***clusters overlap***, however, it becomes necessary to utilize more ***sophisticated techniques*** for ***partitioning the pattern space.*** Overlapping clusters are the result of a ***less*** in observed information and the presence of measurement ***noise***.

The ***basic design concepts*** for automatic pattern recognition described above may be ***implemented by three principal*** categories of ***methodology*** :

1. **Heuristic methods**

   The heuristic approach is based on ***human intuition and experience***, making use of the common-property concepts. A system designed using this principle generally consists of ***a set of ad hoc*** procedures developed for specialized recognition tasks. The structure and performance of a heuristic system will ***depend*** to a large degree on the ***cleverness and experience of the system designers .***

2. **Mathematical methods**

The mathematical approach is based on classification rules which are formulated and derived in a mathematical framework , making use of the common-property and clustering concepts. This is in contrast with the heuristic approach , in which decisions are based on ad hoc rules.

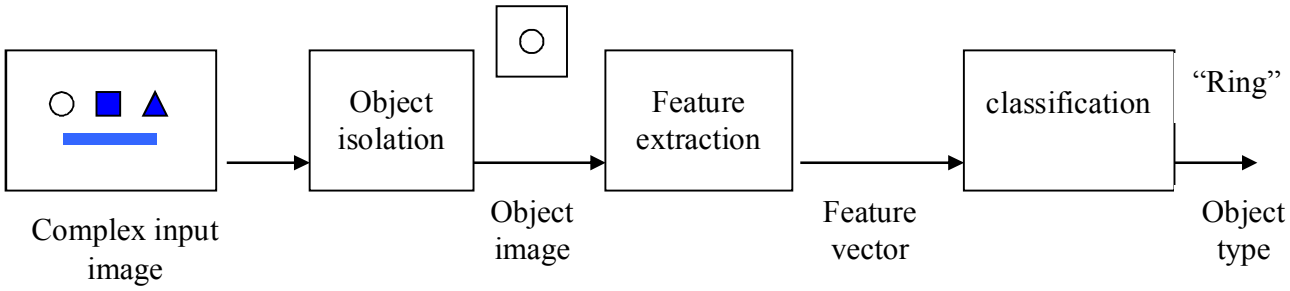
3. Linguistic (syntactic) methods

Characterization of patterns by primitive elements (sub patterens) and their relationships suggest automatic pattern recognition by the linguistic or syntactic approach, making use of the common-property concept. This permits application of formal languages theory to the pattern recognition problem. A pattern grammar is considered as consisting of finite sets of elements called ***variables, primitives, and production. The rules of production determine the type of grammar***. Among the most studied grammars are regular grammars, context-free grammars, and context-sensitive grammar.

12.Optical Pattern Recognition

In this lecture, we address some aspect of analyzing image content. In particular, we shall consider optical pattern recognition implemented by digital image processing techniques. This involve first finding the objects within an image and then identifying (classifying) those objects.

Computer vision is concerned with developing algorithms for analyzing image content. While a variety of approaches toward "image understanding" have been proposed, we shall consider only one, optical pattern recognition. This approach assumes that the image may contain one or more objects and that each object belongs to one of several predetermined type or classes. While optical pattern recognition can be implemented in several different ways, we are concerned only with its implementation by digital image processing techniques. Given a digitized image containing several objects, the pattern recognition process consists of three major phases:



The three phases of pattern recognition

The ***first*** phase is object isolation, in which each object must be found and its image isolated from the rest of the scene. The ***second*** phase is called

feature extraction. The feature is a set of measurable properties. The feature extraction phase measures these properties, producing a set of measurements called the feature vector. This drastically reduced amount of information represents all the knowledge upon which the subsequent classification must be based. The third phase is object classification and its output is merely a decision regarding the class to which the object belongs. The object is thus recognized as being one particular type of objects, and the recognition is implemented as a classification process. Each object is assigned to one of several pre-established groups (classes) that represent the possible type of objects expected to be encountered. The classification is based solely on the feature vector.

1. Pattern Recognition System Design

The design of a pattern recognition systems is usually done in the five steps listed in table below. These are object locator design, feature selection, classifier design, classifier training, and performance evaluation.

The object locator is the algorithm that isolates the image of the individual objects in the complex scene, this is scene segmentation feature selection involve deciding which properties of the object (size, shape , etc) best distinguish among the various object type and thus should be measured. Classification design consist of establishing a mathematical basis for the classification procedure. The various adjustable parameters of the classifier itself (decision threshold , etc) are pinned down in the training stage. Finally , it is usually desirable to estimate the error rates that can be expected with the system. This constitutes the performance evaluation step.

Pattern Recognition System Design

Step	Function
1. Object location design	**Select the scene segmentation algorithm that will isolate the individual objects in the image.**
2. Feature selection	**Decide Which properties of the objects best distinguish the object type and how to measure these.**
3. classification design	**Establish the mathematical basis of the classification algorithm and select the type of classifier structure to be used.**
4. Classifier training	**Fix the various adjustable parameters (decision boundaries ,etc) in the classifier to suit the object being classified.**
5. performance evaluation	**Estimate the expected rates of the various possible misclassification errors.**

2. Patterns and pattern classes

A pattern is a quantative or structural description of an object or some other entity of interest in an image. A pattern class is a family of patterns that share some common properties. Pattern recognition by machine involve techniques for assigning patterns to their respective classes automatically and with as human intervention as possible.

The two principal pattern arrangements used practice are vectors (for quantitative descriptions) and trees (for structural descriptions). Pattern

vectors are represented by lower case letters , such as x,y and z and take the form :

$$X= \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ xn \end{pmatrix}$$

Where each component xi represent the i th descriptor (feature) and n is the number of such descriptors. The nature of the components of a pattern vector x depends on the measurement technique used to describe the physical pattern itself. For example, suppose that we want to describe three type of iris flowers (Iris setosa, virglnica, and versicolor) by measuring the width and length of their petals. In this case we would be dealing with pattern vector of the form

$$X= \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Where x1, x2 correspond to petal length and width, respectively. Because the petals of all flowers vary in width and length to some degree, the pattern vectors describing these flowers also will vary, not only between different classes, but also within a class.

3. ***Feature Selection***

The goal in image analysis is to extract information useful for solving application based problems. This is done by intelligently reducing the amount of image data with the tools we have explored, including edge detection and segmentation. After we have performed these operations, we have modified the image from the lowest level of pixels data into higher-level representations. Now, we can consider extraction of features that can be useful for solving computer imaging problem.

If we desire a system to distinguish objects of different type, we must first decide which parameters, descriptive of the objects, will be measured. The particular parameters that are measured are called the features.

Example

We are working on a machine vision problem for robotic control. We need to control a robotic gripper that picks parts from an assembly line and puts them into boxes. In order to do this we need to determine:

1 – Where the object is in the two – dimensional plane in which the object lie.

2 – What type of object it is – one type goes into box A; another type goes into box B.

First, we define the feature vector that will solve this problem – we determine that knowing the area and center of area of the object, defined by an (r, c) pair, will locate it in space – we determine that if we also know the perimeter we can identify the object. So our feature vector contains four feature measures, we can define it as [area, r, c, perimeter]

Pattern Classification

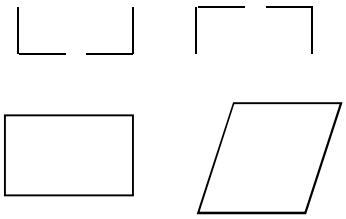
Objects are classified based on the set of feature extracted in the image analysis phase. Key to the image classification task is the fact that similarity between objects implies that they contain similar features, which in turn form the classes.

The primary use of pattern classification in image analysis is for computer vision application development. It can be considered a part of feature analysis, or as a post processing step to feature extraction and analysis. Pattern classification is typically the final step in the development of computer vision algorithm because in these types of applications the goal is to identify objects (or part of objects) in order for computer to perform some vision-related task.

Common pattern classification techniques include template matching, NNC and maximum-Likelihood classifier.

template matching

Template matching involves comparing an ideal representation of a pattern or object , to segmented regions within the image. This can be carried out on a global bases or locally whereby we can use several local features, such as corners (figure below). The key problem with this approach is that it cannot easily deal with scale and/or orientation changes, therefore it can be computationally expensive.



Neares Neighbour classifier (NNC)

Classifies an object based on the closest class that the unknown object lies near. This is also referred to as Maximum Similarity Classifier. This approach is simple to implement and hence popular, although incorrect classification is possible. Assuming an object Q described by the vector $(x_1,x_2,\ldots\ldots\ldots,x_n)$, then a set of m reference vectors, describing objects of each class, denoted by Y_1 , Y_2, ...,Y_m where each vector Y_i= (Y_{i1},Y_{i2}, ….,Y_{in}) similarity between two objects represented X_i and Y_i can be assessed by measuring the Euclidean distance (D (X_i, Yi)) between them .

The larger D(X_i, Y_i) is the smaller between the patterns they represent . Thus, an object of unknown type and which is represented by a vector X can be attributed to an appropriate class by finding which of the Y_1 , Y_2, ...,Y_m values are closet to X. There are many measure (including Euclidean distance) can be used to determine the similarity between two objects.

Maximum-Likelihood Classifier

Assume two closes of objects C1 and C2 , with a single feature to distinguish these class (e.g. , the number of holes, x). This technique involve finding the probability that an object from a given class will have a given feature value. This is found by measuring x for a large number of samples of each class.

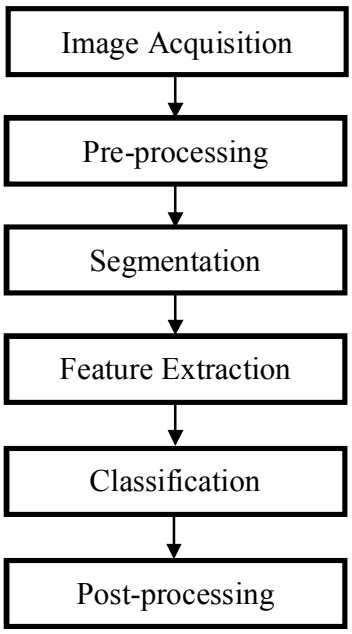
13. OCR and CBIR

OCR (Optical Character Recognition) Systems

An OCR system translates images of text into machine - editable text.

OCR systems consist of four major stages :

- Pre-processing
- Segmentation
- Feature Extraction
- Classification
- Post-processing



1. Pre-processing

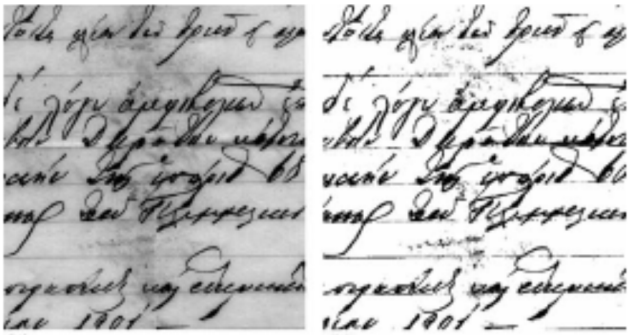
The raw data is subjected to a number of preliminary processing steps to make it usable in the descriptive stages of character analysis. Pre-processing aims to produce data that are easy for the OCR systems to operate accurately. The main objectives of pre-processing are :

- Binarization

- Noise reduction
- Normalization
- Smoothing
- Skew correction

☐ **Document image binarization** (thresholding) refers to the conversion of a gray-scale image into a binary image. Two categories of thresholding:

- Global, picks one threshold value for the entire document image which is often based on an estimation of the background level from the intensity histogram of the image.
- Adaptive (local), uses different values for each pixel according to the local area information

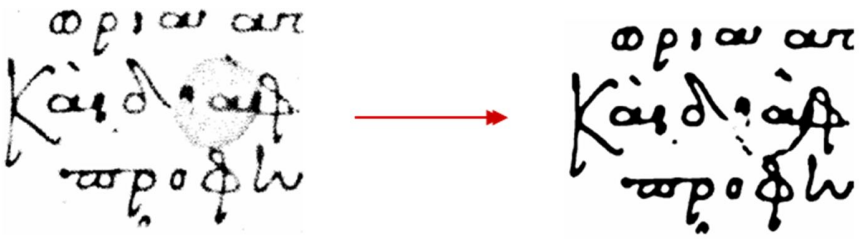


☐ **Noise Reduction ,**The image resulting from the scanning process may contain a certain amount of noise depending on the *resolution on the scanner* and the *success of the technique for thresholding.*

Noise reduction improves the quality of the document. Two main approaches were found:

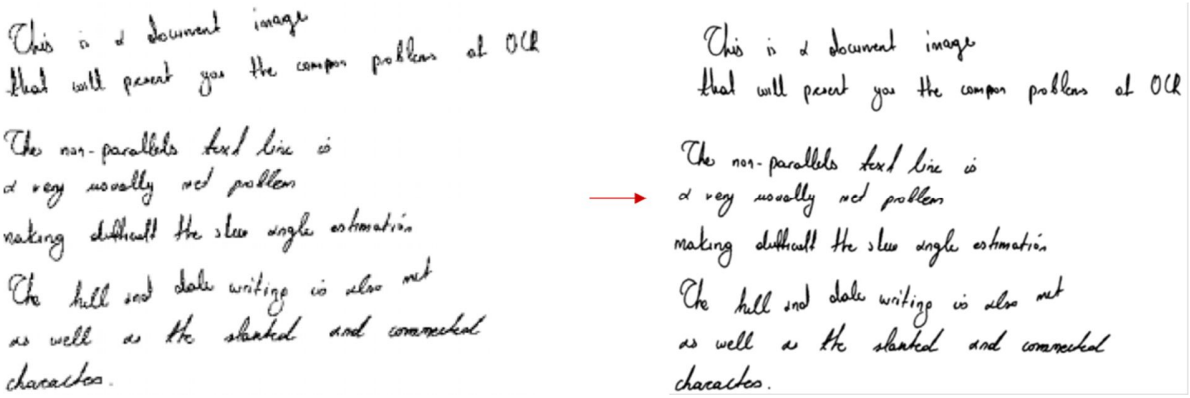
- Filtering (masks)

- Morphological Operations (erosion, dilation, etc)



☐ **Normalization-Smoothing** Normalization provides a large reduction in data size. In order to extract stable feature values, the normalization is applied to obtain characters of uniform size, slant and rotation. The smoothing implies both filling and thinning. Filling eliminates small breaks, gaps and holes in the digitized characters, while thinning reduces the width of the line.

☐ **Skew Correction** methods are used to align the paper document with the coordinate system of the scanner. Main approaches for skew detection include correlation, projection profiles, Hough transform.



2. Segmentation

- **Text segmentation** is the isolation of characters or words. The majority of OCR algorithms segment the words into isolated characters which are recognized individually. Usually this segmentation is performed by isolating each connected component, that is each connected black area. The main problems in segmentation may be divided into four groups:

1- Extraction of touching and fragmented characters.

Such distortions may lead to several joint characters being interpreted as one single character, or that a piece of a character is believed to be an entire symbol.

2- Distinguishing noise from text.

Dots and accents may be mistaken for noise, and vice versa.

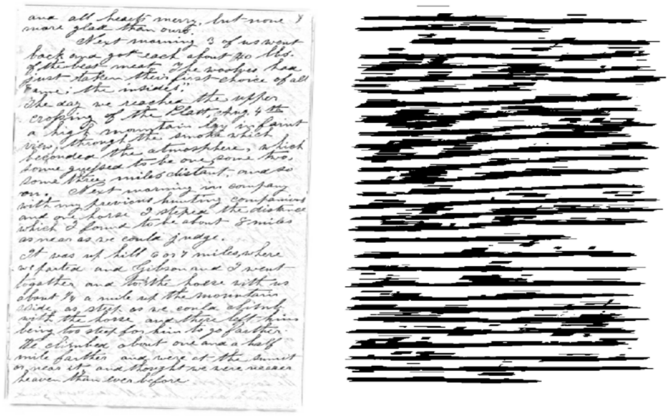
3- Mistaking graphics or geometry for text.

This leads to non text being sent to recognition.

4- Mistaking text for graphics or geometry.

In this case the text will not be passed to the recognition stage. This often happens if characters are connected to graphics.

- Text Line Detection



- Word Extraction

3. Feature Extraction

In feature extraction stage each character is represented as a feature vector, which becomes its identity. The major goal of feature extraction is to extract a set of features, which maximizes the recognition rate with the least amount of elements. Feature extraction methods are based on **statistical** and **structural.**

Statistical Features

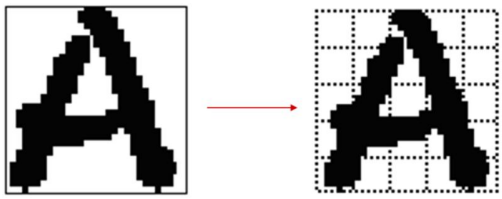
Representation of a character image by statistical distribution of points takes care of style variations to some extent. The major statistical features used for character representation are:

- Zoning

- Projections
- and profile

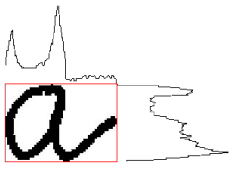
Zoning

The character image is divided into N x M zones. From each zone features are extracted to form the feature vector. The goal of zoning is to obtain the local characteristics instead of global characteristics



Projection Histograms

The basic idea behind using projections is that character images, which are 2-D signals, can be represented as 1-D signal. These features, although independent to noise and deformation, depend on rotation. Projection histograms count the number of pixels in each column and row of a character image. Projection histograms can separate characters such as “m” and “n” .



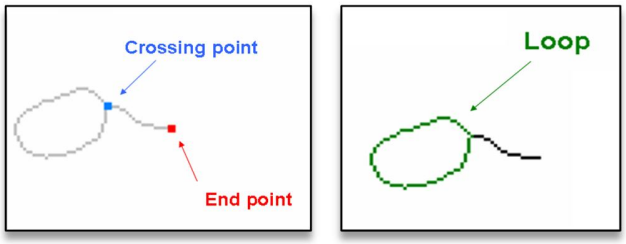
Profiles

The profile counts the number of pixels (distance) between the bounding box of the character image and the edge of the character.

The profiles describe well the external shapes of characters and allow to distinguish between a great number of letters, such as “p” and “q”.

Structural Features

Characters can be represented by structural features with high tolerance to distortions and style variations. Structural features are based on topological and geometrical properties of the character, such as aspect ratio, cross points, loops, branch points, strokes and their directions, inflection between two points, horizontal curves at top or bottom, etc.

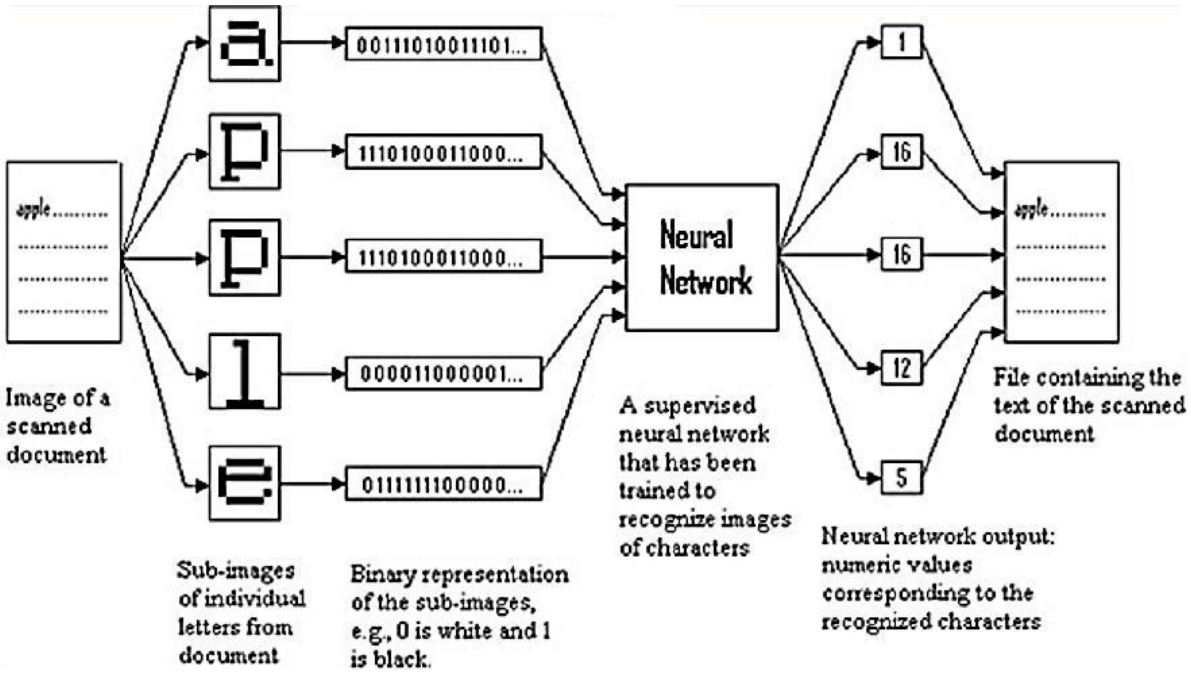


4. Classification

The classification is the process of identifying each character and assigning to it the correct character class. There is no such thing as the “best classifier”. The use of classifier depends on many factors, such as available training set, number of free parameters etc.

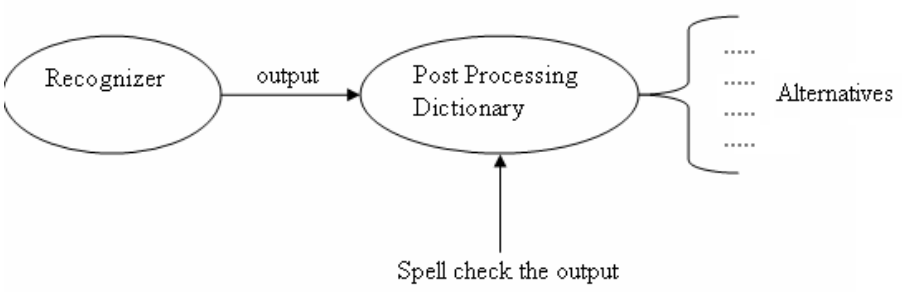
Neural networks

- A feature vector enters the network at the input layer. Each element of the layer computes a weighted sum of its input and transforms it into an output by a nonlinear function. During training the weights at each connection are adjusted until a desired output is obtained.



5. Post-processing

The incorporation of context and shape information in all the stages of OCR systems is necessary for meaningful improvements in recognition rates. The simplest way of incorporating the context information is the utilization of a dictionary for correcting the minor mistakes.



Content-based Image Retrieval (CBIR)

There are two approaches to image retrieval: Text-Based approach and Content- Based approach. Today, the most common way of doing this is by textual descriptions and categorizing of images. Different people might categorize or describe the same image differently, leading to problems retrieving it again. It is also time consuming when dealing with very large databases. CBIR is a way to get around these problems.

CBIR systems search collection of images based on features that can be extracted from the image files themselves without manual descriptive. Comparing two images and deciding if they are similar or not is a relatively easy thing to do for a human. Getting a computer to do the same thing effectively is however a different matter. Many different approaches to CBIR have been tried and many of these have one thing in common, the use of color histograms.

CBIR is suitable for medical diagnoses based on the comparison of X-ray pictures with past cases, and for finding the faces of criminals from video shots of a crowd. CBIR systems use visual content such as color, texture, and simple shape properties to search images from large scale image databases.

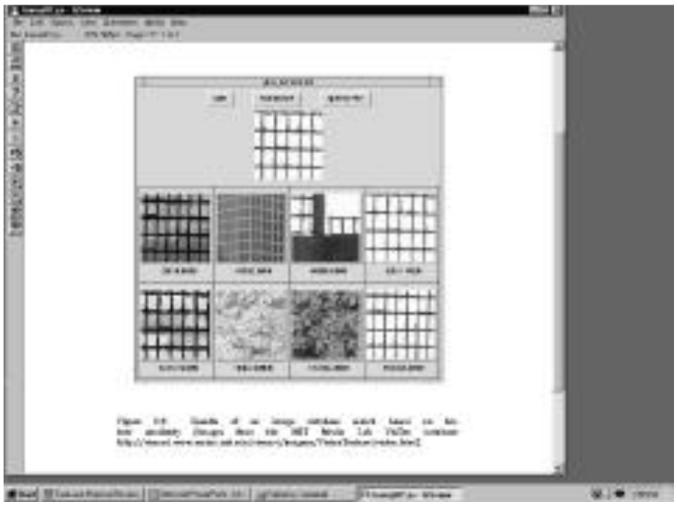
Some of the more commonly used types of feature used for image retrieval are described below:

1. Colour retrieval

Several methods for retrieving images on the basis of colour similarity have been described in the literature, but most are variations on the same basic idea. Each image added to the collection is analysed to compute a *colour histogram* which shows the proportion of pixels of each colour within the image. The colour histogram for each image is then stored in the database. At search time, the user can either specify the desired proportion of each colour or submit an example image from which a colour histogram is calculated. Either way, the matching process then retrieves those images whose colour histograms match those of the query most closely.

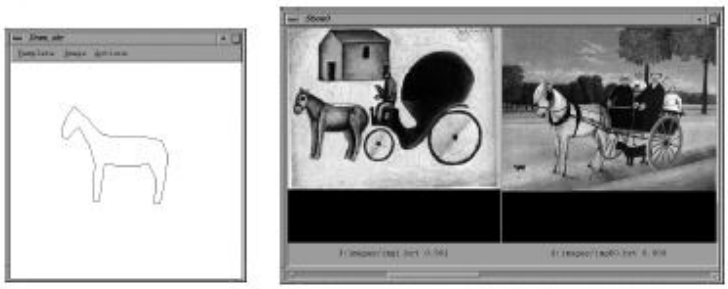
2. Texture retrieval

The ability to retrieve images on the basis of texture similarity may not seem very useful. But the ability to match on texture similarity can often be useful in distinguishing between areas of images with similar colour (such as sky and sea, or leaves and grass). A variety of techniques has been used for measuring texture similarity; the best-established rely on comparing values of what are known as *second-order statistics* calculated from query and stored images. Essentially, these calculate the relative brightness of selected *pairs* of pixels from each image. From these it is possible to calculate measures of image texture such as the degree of *contrast, coarseness (خشونة)*, r*egularity*, *periodicity (دوريا)* and *randomness*.

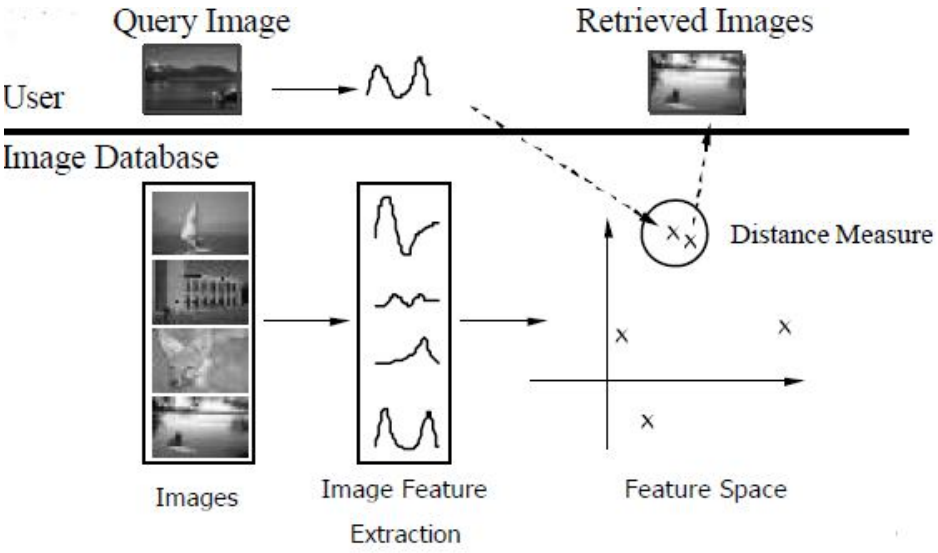


3. Shape retrieval

The ability to retrieve by shape is perhaps the most obvious requirement at the primitive level. Unlike texture, shape is a fairly well-defined concept – and there is considerable evidence that natural objects are primarily recognized by their shape. A number of features characteristic of object shape (but independent of size or orientation) are computed for every object identified within each stored image. Queries are then answered by computing the same set of features for the query image, and retrieving those stored images whose features most closely match those of the query. Two main types of shape feature are commonly used – *global* features such as aspect ratio, circularity and moment invariants and *local* features such as sets of consecutive boundary segments.



We can summarize the CBIR system by the figure below:



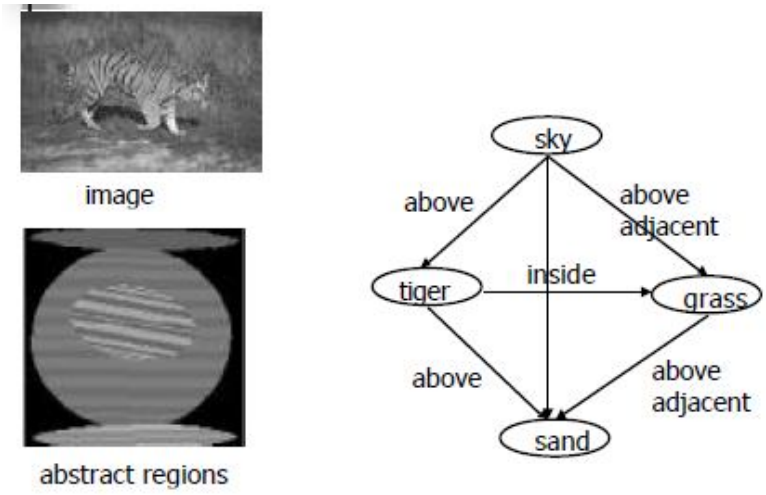
Regions and Relationships

• Segment the image into regions.

• Find their properties and interrelationships.

• Construct a graph representation with nodes for regions and edges for spatial relationships.

• Use graph matching to compare images.



Tiger Image as a Graph

LECTURE 14

EDGE LINE DETECTION

1. Define of edge / line detection

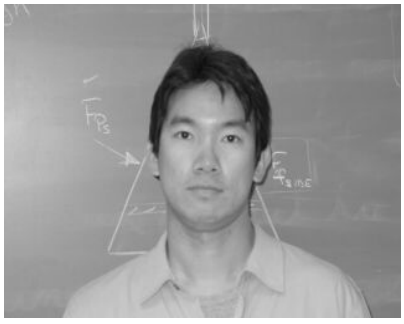
The edge and line detection operators presented here represent the various types of operators used today. They are implemented with convolution masks and most are based on discrete approximations to differential operators. Edge detection methods are used as a first step in the line detection process.

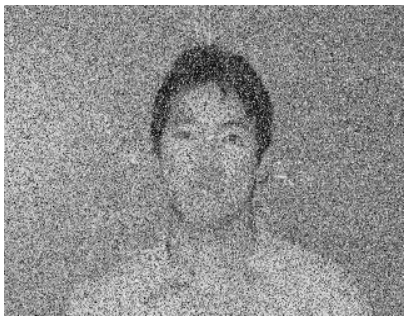
Detecting edges is a basic operation in image processing. The edges of items in an image hold much of the information in the image. The edges tell you where:

- 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.

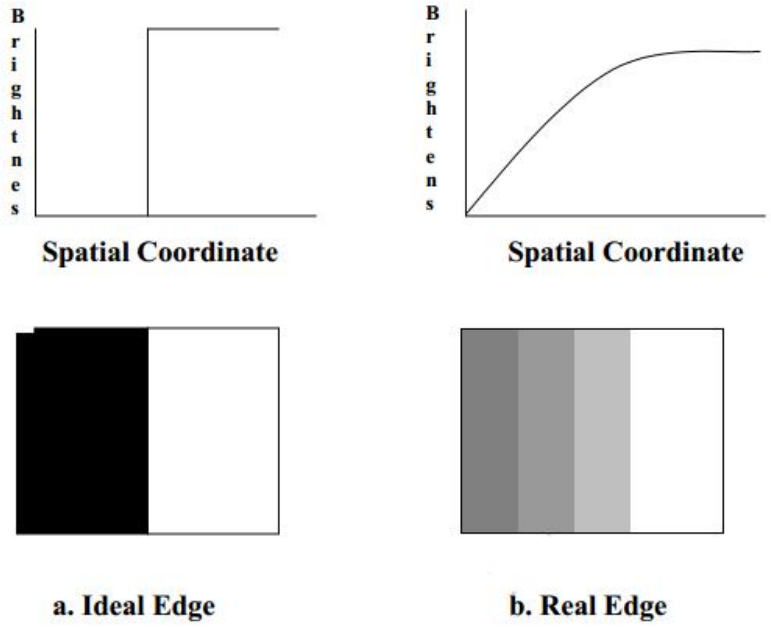




2. Methods of detection edge / line

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



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:

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

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

P1	P2
P3	P4

7	9	44	4
2	4	7	120

9	10	11	12
13	14	15	16

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

New-pixel= [(p1 – p4) +(p2-p3)] then

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

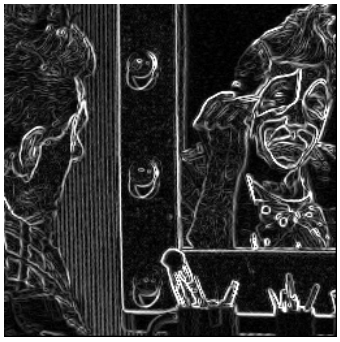
7	9	44	4
2	4	7	120
9	10	11	12
13	14	15	16

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

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

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

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

Projection on the horizontal masks

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

New – pixel =

P1	**p2**	**p3**
p4	p5	p6
p7	p8	p9

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

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

Edge magnitude =

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

= 109*109+125*125=165.8

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

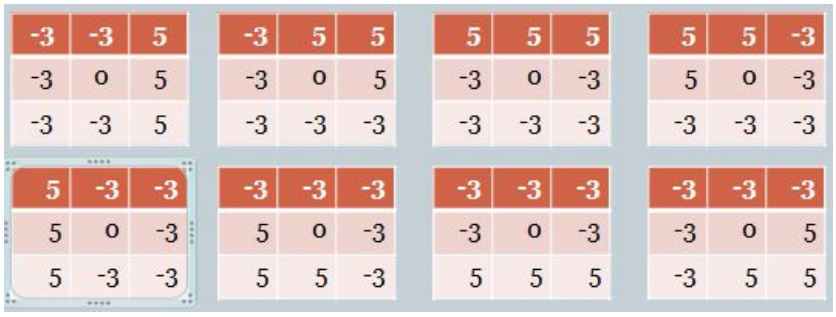
The same of sobel but

-1	0	1
-1	0	1
-1	0	1

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.



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.

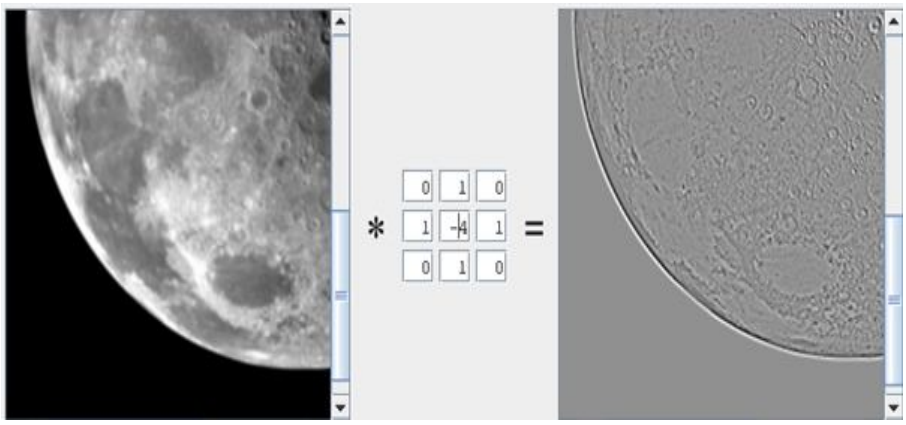
0	**-1**	**0**
-1	4	-1
0	-1	0

1	**-2**	**1**
-2	4	-2
1	-2	1

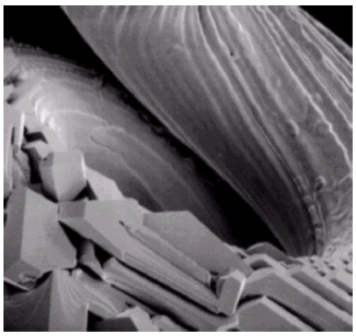
-1	**-1**	**-1**
-1	8	-1
-1	-1	-1

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.

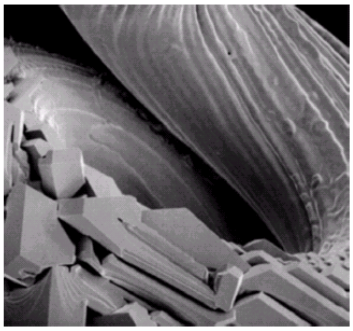
For example see bellow



Otherwise we can use the Laplacian in enhancement for example



0	**-1**	**0**
-1	4	-1
0	-1	0



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

$$\frac{1}{2\sqrt{2}}\begin{bmatrix}1 & \sqrt{2} & 1\\ 0 & 0 & 0\\ -1 & -\sqrt{2} & -1\end{bmatrix}\quad f1 \qquad \frac{1}{2\sqrt{2}}\begin{bmatrix}1 & 0 & -1\\ \sqrt{2} & 0 & -\sqrt{2}\\ 1 & 0 & -1\end{bmatrix}\quad f2 \qquad \frac{1}{2\sqrt{2}}\begin{bmatrix}0 & -1 & \sqrt{2}\\ 1 & 0 & -1\\ -\sqrt{2} & 1 & 0\end{bmatrix}\quad f3$$

$$\frac{1}{2\sqrt{2}}\begin{bmatrix}\sqrt{2} & -1 & 0\\ -1 & 0 & 1\\ 0 & 1 & -\sqrt{2}\end{bmatrix}\quad f4 \qquad \frac{1}{2}\begin{bmatrix}0 & 1 & 0\\ -1 & 0 & -1\\ 0 & 1 & 0\end{bmatrix}\quad f5 \qquad \frac{1}{2}\begin{bmatrix}-1 & 0 & 1\\ 0 & 0 & 0\\ 1 & 0 & -1\end{bmatrix}\quad f6$$

$$\frac{1}{6}\begin{bmatrix}1 & -2 & 1\\ -2 & 4 & -2\\ 1 & -2 & 1\end{bmatrix}\quad f7 \qquad \frac{1}{6}\begin{bmatrix}-2 & 1 & -2\\ 1 & 4 & 1\\ -2 & 1 & -2\end{bmatrix}\quad f8 \qquad \frac{1}{3}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}\quad f9$$

Example:

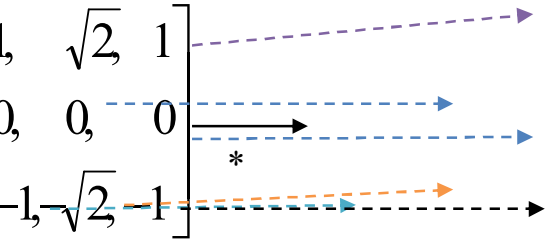
W

1	0	1

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

1	0	1
1	0	1
1	0	1

For (f1) = $1/2\sqrt{2}\begin{bmatrix}1, & \sqrt{2}, & 1\\ 0, & 0, & 0\\ -1, & -\sqrt{2}, & -1\end{bmatrix}$ *



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

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

f1 →0, f2 →0, f3 →0, f4 →0, f5 →-1, f6 →0, f7 →0, f8 →-1, f9 →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.

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

*I*s

Then used the information to find the edge by

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

Set {e} consists of the masks of interest. The (Is, 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×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×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,

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

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