GOVERNMENT ARTS AND SCIENCE COLLEGE

KOMARAPALAYAM-638 183

CLASS : II M.Sc COMPUTER SCIENCE

SEMESTER : III

SUBJECT NAME: DIGITAL IMAGE PROCESSING

SUBJECT CODE : 19PCS12

HANDLED BY

1.DR.P.R.TAMILSELVI

2.DR.M.GOMATHI

UNIT-I

INTRODUCTION:

An image may be defined as a two-dimensional function, f(x, y), where *x* and *y* are *spatial* (plane) coordinates, and the amplitude of *f* at any pair of coordinates (*x*, *y*) is called the *intensity* or *gray level* of the image at that point. When *x*, *y*, and the intensity values of *f* are all finite, discrete quantities, we call the image a *digital image*.

The field of *digital image processing* refers to processing digital images by means of a digital computer. Note that a digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are called *picture elements*, *image elements*, *pels*, and *pixels*. *Pixel* is the term used most widely to denote the elements of a digital image.

Vision is the most advanced of our senses, so it is not surprising that images play the single most important role in human perception. However, unlike humans, who are limited to the visual band of the electromagnetic (EM) spectrum, imaging machines cover almost the entire EM spectrum, ranging from gamma to radio waves. They can operate on images generated by sources that humans are not accustomed to associating with images.

These include ultrasound, electron microscopy, and computer-generated images. Thus, digital image processing encompasses a wide and varied field of applications. Sometimes a distinction is made by defining image processing as a discipline in which both the input and output of a process are images.

For example, under this definition, even the trivial task of computing the average intensity of an image (which yields a single number) would not be considered an image processing operation. On the other hand, there are fields such as computer vision whose ultimate goal is to use computers to emulate human vision, including learning and being able to make inferences and take actions based on visual inputs. This area itself is a branch of artificial intelligence (AI) whose objective is to emulate human intelligence.

The field of AI is in its earliest stages of infancy in terms of development, with progress having been much slower than originally anticipated. The area of image analysis (also called image understanding) is in between image processing and computer vision. There are no clear-cut boundaries in the continuum from image processing at one end to computer vision at the other.

However, one useful paradigm is to consider three **types of computerized processes** in this continuum:

- low- level processes
- mid- level processes
- high-level processes

Low-level processes involve primitive operations such as image preprocessing to reduce noise, contrast enhancement, and image sharpening. A low-level process is characterized by the fact that both its inputs and outputs are images.

Mid-level processing on images involves tasks such as segmentation (partitioning an image into regions or objects), description of those objects to reduce them to a form suitable for computer processing, and classification (recognition) of individual objects. A mid-level process is characterized by the fact that its inputs generally are images, but its outputs are attributes extracted from those images (e.g., edges, contours, and the identity of individual objects).

High-level processing involves "making sense" of an ensemble of recognized objects, as in image analysis, and, at the far end of the continuum, performing the cognitive functions normally associated with vision.

Based on the preceding comments, we see that a logical place of overlap between image processing and image analysis is the area of recognition of individual regions or objects in an image. *Digital image processing* encompasses processes whose inputs and outputs are images and, in addition, encompasses processes that extract attributes from images, up to and including the recognition of individual objects.

Making sense of the content of the page may be viewed as being in the domain of image analysis and even computer vision, depending on the level of complexity implied by the statement "making sense."As will become evident shortly, digital image processing, as we have defined it, is used successfully in a broad range of areas of exceptional social and economic value. The concepts developed in the following chapters are the foundation for the methods used in those application areas.

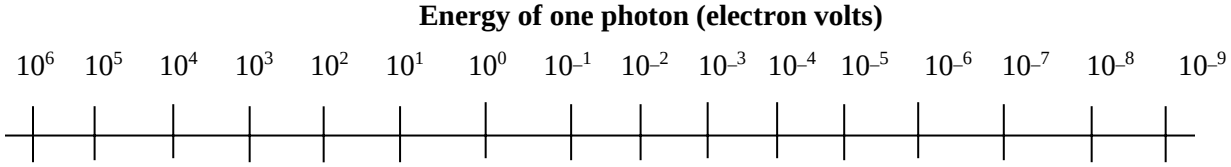
EXAMPLES OF FIELDS THAT USE DIGITAL IMAGE PROCESSING:

The areas of application of digital image processing are so varied that some form of organization is desirable in attempting to capture the breadth of this field. One of the simplest ways to develop a basic understanding of the extent of image processing applications is to categorize images according to their source (e.g., visual, X-ray, and so on).

The principal energy source for images in use today is the electromagnetic energy spectrum. Other important sources of energy include acoustic, ultrasonic, and electronic (in the form of electron beams used in electron microscopy). Synthetic images, used for modeling and visualization, are generated by computer. In this section we discuss briefly how images are generated in these various categories and the areas in which they are applied. Methods for converting images into digital form are discussed in the next chapter.

Images based on radiation from the EM spectrum are the most familiar, especially images in the X-ray and visual bands of the spectrum. Electromagnetic waves can be conceptualized as propagating sinusoidal waves of varying wavelengths, or they can be thought of as a stream of massless particles, each traveling in a wavelike pattern and moving at the speed of light.

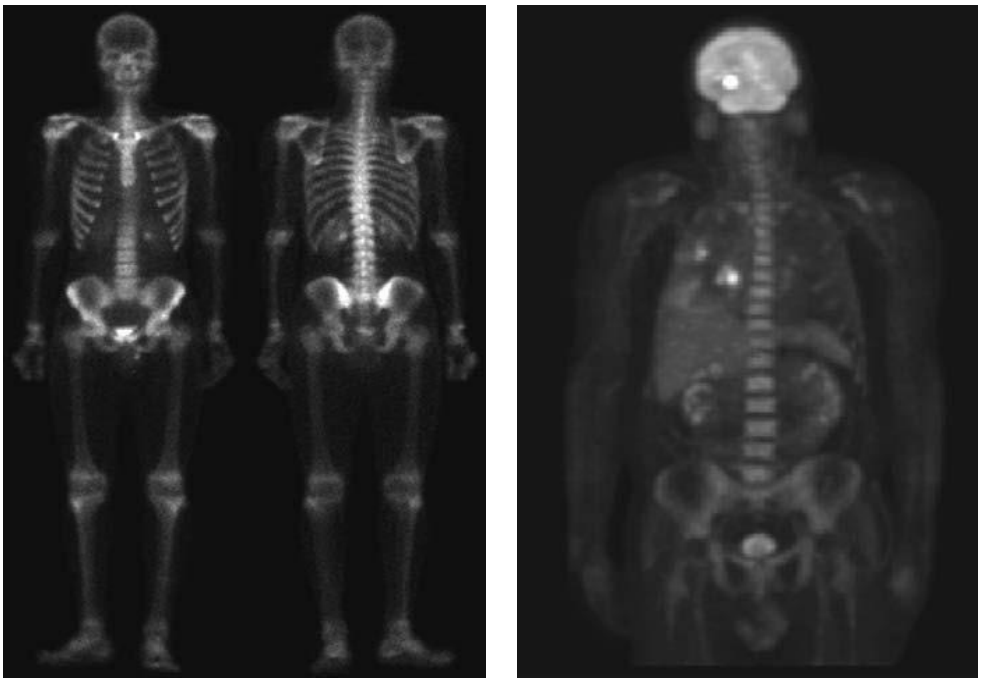
Each massless particle contains a certain amount (or bundle) of energy. Each bundle of energy is called a *photon*. If spectral bands are grouped according to energy per photon, we obtain the spectrum shown in Figure, ranging from gamma rays (highest energy) at one end to radio waves (lowest energy) at the other.



The bands are shown shaded to convey the fact that bands of the EM spectrum are not distinct but rather transition smoothly from one to the other.

Gamma-Ray Imaging:

Major uses of imaging based on gamma rays include nuclear medicine and astronomical observations. In nuclear medicine, the approach is to inject a patient with a radioactive isotope that emits gamma rays as it decays. Images are produced from the emissions collected by gamma ray detectors. Figure (a) shows an image of a complete bone scan obtained by using gamma-ray imaging. Images of this sort are used to locate sites of bone pathology, such as infections or tumors.



(a) (b)

Figure (b) shows another major modality of nuclear imaging called positron emission tomography (PET). However, instead of using an external source of X-ray energy, the patient is given a radioactive isotope that emits positrons as it decays. When a positron meets an electron, both are annihilated and two gamma rays are given off. These are detected and a tomographic image is created using the basic principles of tomography.

The image shown in Fig.(b) is one sample of a sequence that constitutes a 3-D rendition of the patient. This image shows a tumor in the brain and one in the lung, easily visible as small white masses. A star in the constellation of Cygnus exploded about 15,000 years ago, generating a superheated stationary gas cloud (known as the Cygnus Loop) that glows in a spectacular array of colors left side of the image.

X- Ray Imaging:

X-rays are among the oldest sources of EM radiation used for imaging. The best known use of X-rays is medical diagnostics, but they also are used extensively in industry and other areas, like astronomy. X-rays for medical and industrial imaging are generated using an X-ray tube, which is a vacuum tube with a cathode and anode. The cathode is heated, causing free electrons to be released. These electrons flow at high speed to the positively charged anode.

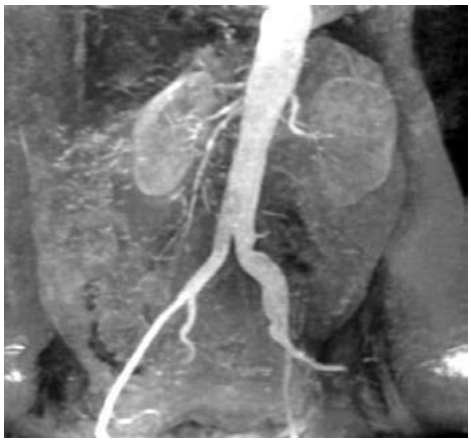
When the electrons strike a nucleus, energy is released in the form of X-ray radiation. The energy (penetrating power) of X-rays is controlled by a voltage applied across the anode, and by a current applied to the filament in the cathode. Figure (a) shows a familiar chest X-ray generated simply by placing the patient between an X-ray source and a film sensitive to X-ray energy.

The intensity of the X-rays is modified by absorption as they pass through the patient, and the resulting energy falling on the film develops it, much in the same way that light develops photographic film. In digital radiography, digital images are obtained by one of two methods:

(1) By digitizing X-ray films; or

(2)By having the X-rays that pass through the patient fall directly onto devices (such as a phosphor screen) that convert X-rays to light. The light signal in turn is captured by a light-sensitive digitizing system. Angiography is another major application in an area called contrast enhancement radiography. This procedure is used to obtain images (called *angiograms*) of blood vessels.

A catheter (a small, flexible, hollow tube) is inserted; for example, into an artery or vein in the groin. The catheter is threaded into the blood vessel and guided to the area to be studied. When the catheter reaches the site under investigation, an X-ray contrast medium is injected through the tube. This enhances contrast of the blood vessels and enables the radiologist to see any irregularities or blockages. Figure (b) shows an example of an aortic angiogram. The catheter can be seen being inserted into the large blood vessel on the picture.



(a) (b)

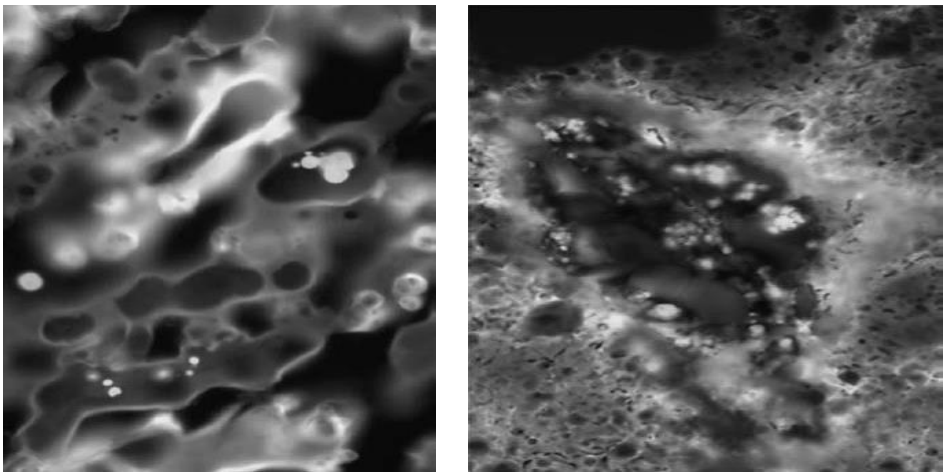
Imaging in the Ultraviolet Band:

Applications of ultraviolet "light" are varied. They include lithography, industrial inspection, microscopy, lasers, biological imaging, and astronomical observations. Ultraviolet light is used in fluorescence microscopy, one of the fastest growing areas of microscopy.

Fluorescence is a phenomenon discovered in the middle of the nineteenth century, when it was first observed that the mineral fluorspar fluoresces when ultraviolet light is directed upon it. The ultraviolet light itself is not visible, but when a photon of ultraviolet radiation collides with an electron in an atom of a fluorescent material, it elevates the electron to a higher energy level. Subsequently, the excited electron relaxes to a lower level and emits light in the form of a lower-energy photon in the visible (red) light region.

The basic task of the fluorescence microscope is to use an excitation light to irradiate a prepared specimen and then to separate the much weaker radiating fluorescent light from the brighter excitation light. Thus, only the emission light reaches the eye or other detector. The resulting fluorescing areas shine against a dark background with sufficient contrast to permit detection

Figure (a) shows a fluorescence microscope image of normal corn, and Fig. (b) Shows corn infected by "smut," a disease of cereals, corn grasses, onions, and sorghum that can be caused by any of more than 700 species of parasitic fungi. Corn smut is particularly harmful because corn is one of the principal food sources in the world.



(a) (b)

Imaging in the Visible and Infrared Bands:

Considering that the visual band of the electromagnetic spectrum is the most familiar in all our activities, it is not surprising that imaging in this band outweighs by far all the others in terms of breadth of application. The infrared band often is used in conjunction with visual imaging.

Thematic bands in NASA's LANDSAT satellite:

Band No.	Name	Wavelength (_m)	Characteristics and Uses
1	Visible blue	0.45–0.52	Maximum water penetration
2	Visible green	0.52–0.60	Good for measuring plant vigor
3	Visible red	0.63–0.69	Vegetation discrimination
4	Near infrared	0.76–0.90	Biomass and shoreline mapping
5	Middle infrared	1.55–1.75	Moisture content of soil and vegetation
6	Thermal infrared	10.4–12.5	Soil moisture; thermal mapping
7	Middle infrared	2.08–2.35	Mineral mapping

The primary function of LANDSAT is to obtain and transmit images of the Earth from space for purposes of monitoring environmental conditions on the planet. The bands are expressed in terms of wavelength, with 1μm being equal to 10^{-6}m

Imaging in the Microwave Band:

The dominant application of imaging in the microwave band is radar. The unique feature of imaging radar is its ability to collect data over virtually any region at any time, regardless of weather or ambient lighting conditions.

Radar waves can penetrate clouds, and under certain conditions can also see through vegetation, ice, and dry sand. In many cases, radar is the only way to explore inaccessible regions of the Earth's surface. Imaging radar works like a flash camera in that it provides its own illumination (microwave pulses) to illuminate an area on the ground and take a snapshot image.

Imaging in the Radio Band:

In medicine, radio waves are used in magnetic resonance imaging (MRI). This technique places a patient in a powerful magnet and passes radio waves through his or her body in short pulses. Each pulse causes a responding pulse of radio waves to be emitted by the patient's tissues.

The location from which these signals originate and their strength is determined by a computer, which produces a two-dimensional picture of a section of the patient. MRI can produce pictures in any plane.

Examples in which Other Imaging Modalities Are Used:

Imaging using "sound" finds application in geological exploration, industry, and medicine. Geological applications use sound in the low end of the sound spectrum (hundreds of Hz) while imaging in other areas use ultrasound (millions of Hz).The most important commercial applications of image processing in geology are in mineral and oil exploration.

For image acquisition over land, one of the main approaches is to use a large truck and a large flat steel plate. The plate is pressed on the ground by the truck, and the truck is vibrated through a frequency spectrum up to 100 Hz. The strength and speed of the returning sound waves are determined by the composition of the Earth below the surface. These are analyzed by computer, and images are generated from the resulting analysis.

Although ultrasound imaging is used routinely in manufacturing, the best known applications of this technique are in medicine, especially in obstetrics, where unborn babies are imaged to determine the health of their development. A byproduct of this examination is determining the sex of the baby. Ultrasound images are generated using the following basic procedure:

- The ultrasound system (a computer, ultrasound probe consisting of a source and receiver, and a display) transmits high-frequency (1 to 5 MHz) sound pulses into the body.

- The sound waves travel into the body and hit a boundary between tissues (e.g., between fluid and soft tissue, soft tissue and bone). Some of the sound waves are reflected back to the probe, while some travel on further until they reach another boundary and get reflected.
- The reflected waves are picked up by the probe and relayed to the computer.
- The machine calculates the distance from the probe to the tissue or organ boundaries using the speed of sound in tissue (1540 m/s) and the time of each echo's return.
- The system displays the distances and intensities of the echoes on the screen, forming a two-dimensional image.

FUNDAMENTAL STEPS IN DIGITAL IMAGE PROCESSING:

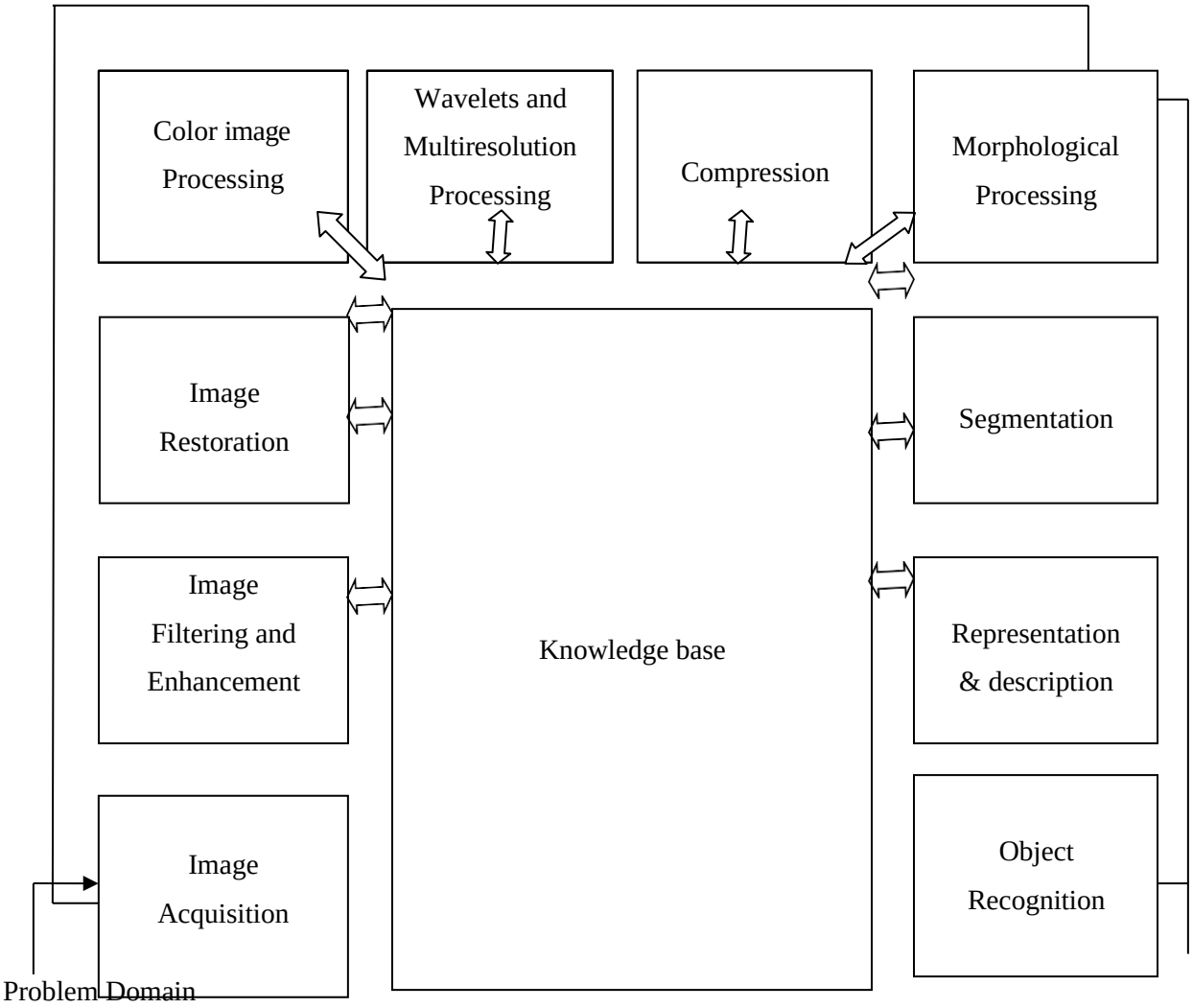
An idea of all the methodologies that can be applied to images for different purposes and possibly with different objectives

Image acquisition is the first process in Figure. Acquisition could be as simple as being given an image that is already in digital form. Generally, the image acquisition stage involves preprocessing, such as scaling.

Image enhancement is the process of manipulating an image so that the result is more suitable than the original for a specific application. The word *specific* is important here, because it establishes at the outset that enhancement techniques are problem oriented. Thus, for example, a method that is quite useful for enhancing X-ray images may not be the best approach for enhancing satellite images taken in the infrared band of the electromagnetic spectrum.

Image restoration is an area that also deals with improving the appearance of an image. However, unlike enhancement, which is subjective, image restoration is objective, in the sense that restoration techniques tend to be based on mathematical or probabilistic models of image degradation. Enhancement, on the other hand, is based on human subjective preferences regarding what constitutes a "good" enhancement result.

Outputs of these processes generally are images



Color image processing is an area that has been gaining in importance because of the significant increase in the use of digital images over the Internet. Color is used also in later chapters as the basis for extracting features of interest in an image.

Wavelets are the foundation for representing images in various degrees of resolution. In particular, this material is used in this book for image data compression and for pyramidal representation, in which images are subdivided successively into smaller regions.

Compression, as the name implies, deals with techniques for reducing the storage required saving an image, or the bandwidth required transmitting it. Although storage technology has improved significantly over the past decade, the same cannot be said for transmission capacity. This is true particularly

in uses of the Internet, which are characterized by significant pictorial content. Image compression is familiar (perhaps inadvertently) to most users of computers in the form of image file extensions, such as the jpg file extension used in the JPEG (Joint Photographic Experts Group) image compression standard.

Morphological processing deals with tools for extracting image components that are useful in the representation and description of shape. a transition from processes that output images to processes that output image attributes

.

Segmentation procedures partition an image into its constituent parts or objects. In general, autonomous segmentation is one of the most difficult tasks in digital image processing. A rugged segmentation procedure brings the process a long way toward successful solution of imaging problems that require objects to be identified individually.

On the other hand, weak or erratic segmentation algorithms almost always guarantee eventual failure. In general, the more accurate the segmentation, the more likely recognition is to succeed.

Representation and description almost always follow the output of a segmentation stage, which usually is raw pixel data, constituting either the boundary of a region (i.e., the set of pixels separating one image region from another) or all the points in the region itself. In either case, converting the data to a form suitable for computer processing is necessary. The first decision that must be made is whether the data should be represented as a boundary or as a complete region.

Boundary representation is appropriate when the focus is on external shape characteristics, such as corners and inflections. Regional representation is appropriate when the focus is on internal properties, such as texture or skeletal shape. In some applications, these representations complement each other. Choosing a representation is only part of the solution for transforming raw data into a form suitable for subsequent computer processing.

A method must also be specified for describing the data so that features of interest are highlighted. ***Description***, also called ***feature selection***, deals with extracting attributes that result in some quantitative information of interest or are basic for differentiating one class of objects from another.

Recognition is the process that assigns a label (e.g., “vehicle”) to an object based on its descriptors. As detailed in Section 1.1, we conclude our coverage of digital image processing with the development of methods for recognition of individual objects.

Need for prior knowledge or about the interaction between the ***knowledge base*** and the processing modules in Figure Knowledge about a problem domain is coded into an image processing system in the form of a knowledge database. This knowledge may be as simple as detailing regions of an image where

the information of interest is known to be located, thus limiting the search that has to be conducted in seeking that information.

The knowledge base also can be quite complex, such as an interrelated list of all major possible defects in a materials inspection problem or an image database containing high-resolution satellite images of a region in connection with change-detection applications. In addition to guiding the operation of each processing module, the knowledge base also controls the interaction between modules. This distinction is made in Figure by the use of double-headed arrows between the processing modules and the knowledge base, as opposed to single headed arrows linking the processing modules.

Although we do not discuss image display explicitly at this point, it is important to keep in mind that viewing the results of image processing can take place at the output of any stage in Figure. We also note that not all image processing applications require the complexity of interactions implied by Figure.

In fact, not even all those modules are needed in many cases. For example, image enhancement for human visual interpretation seldom requires use of any of the other stages in Figure. In general, however, as the complexity of image processing task increases, so does the number of processes required to solve the problem.

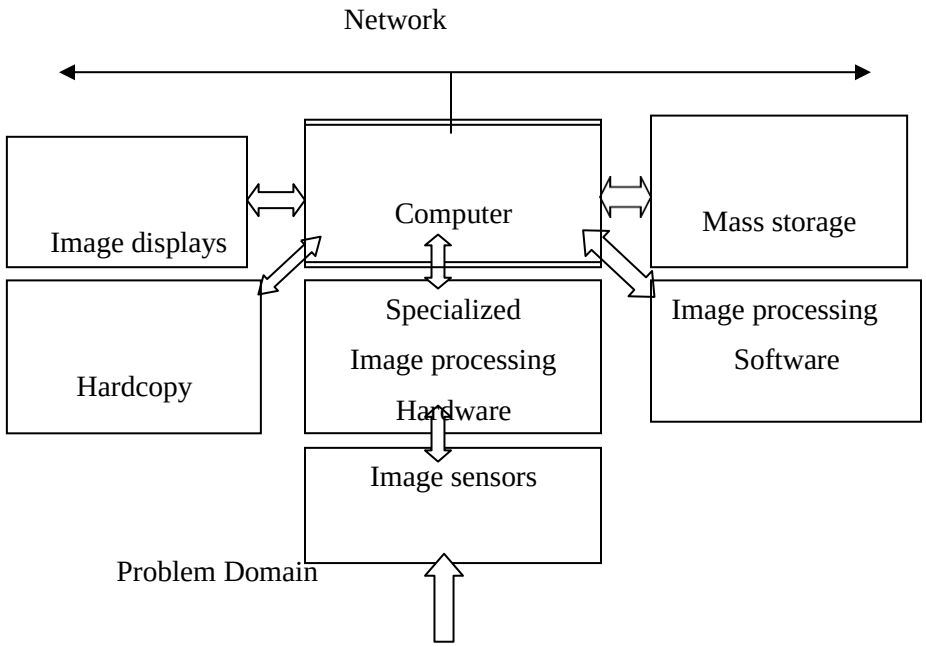
COMPONENTS OF AN IMAGE PROCESSING SYSTEM:

Figure shows the basic components comprising a typical *general-purpose* system used for digital image processing. The function of each component is discussed in the following paragraphs, starting with image sensing. With reference to *sensing*, two elements are required to acquire digital images.

The first is a physical device that is sensitive to the energy radiated by the object we wish to image. The second, called a ***digitizer***, is a device for converting the output of the physical sensing device into digital form. For instance, in a digital video camera, the sensors produce an electrical output proportional to light intensity. The digitizer converts these outputs to digital data.

Specialized image processing *hardware* usually consists of the digitizer just mentioned, plus hardware that performs other primitive operations, such as an arithmetic logic unit (ALU) that performs arithmetic and logical operations in parallel on entire images. One example of how an ALU is used is in averaging images as quickly as they are digitized, for the purpose of noise reduction. This type of hardware sometimes is called a ***front-end subsystem***, and its most distinguishing characteristic is speed. In other words, this unit performs functions that require fast data throughputs (e.g., digitizing and averaging video images at 30 frames/s) that the typical main computer cannot handle.

The ***computer*** in an image processing system is a general-purpose computer and can range from a PC to a supercomputer. In dedicated applications, sometimes custom computers are used to achieve a required level of performance, but our interest here is on general-purpose image processing systems.



In these systems, almost any well-equipped PC-type machine is suitable for off-line image processing tasks.

Software for image processing consists of specialized modules that perform specific tasks. A well-designed package also includes the capability for the user to write code that, as a minimum, utilizes the specialized modules. More sophisticated software packages allow the integration of those modules and general-purpose software commands from at least one computer language.

Mass storage capability is a must in image processing applications. An image of size 1024 X 1024 pixels, in which the intensity of each pixel is an 8-bit quantity, requires one megabyte of storage space if the image is not compressed. When dealing with thousands, or even millions, of images, providing adequate storage in an image processing system can be a challenge. Digital storage for image processing applications falls into three principal categories:

(1) Short-term storage for use during processing.

(2) On-line storage for relatively fast recall.

(3) Archival storage, characterized by infrequent access.

Storage is measured in bytes (eight bits), Kbytes (one thousand bytes), Mbytes (one million bytes), Gbytes (meaning giga, or one billion, bytes), and Tbytes (meaning tera, or one trillion, bytes). One method of providing short-term storage is computer memory.

Another is by specialized boards, called ***frame buffers***, that store one or more images and can be accessed rapidly, usually at video rates (e.g., at 30 complete images per second). The latter method allows virtually instantaneous image ***zoom,*** as well as ***scroll*** (vertical shifts) and ***pan*** (horizontal shifts).

Frame buffers usually are housed in the specialized image processing hardware unit in Figure. On-

line storage generally takes the form of magnetic disks or optical-media storage. The key factor characterizing on-line storage is frequent access to the stored data. Finally, archival storage is characterized by massive storage requirements but infrequent need for access.

Image displays in use today are mainly color (preferably flat screen) TV monitors. Monitors are driven by the outputs of image and graphics display cards that are an integral part of the computer system. Seldom are there requirements for image display applications that cannot be met by display cards available commercially as part of the computer system.

Hardcopy devices for recording images include laser printers, film cameras, heat-sensitive devices, inkjet units, and digital units, such as optical and CDROM disks. Film provides the highest possible resolution, but paper is the obvious medium of choice for written material. For presentations, images are displayed on film transparencies or in a digital medium if image projection equipment is used. The latter approach is gaining acceptance as the standard for image presentations.

Networking is almost a default function in any computer system in use today. Because of the large amount of data inherent in image processing applications, the key consideration in image transmission is bandwidth. In dedicated networks, this typically is not a problem, but communications with remote sites via the Internet are not always as efficient.

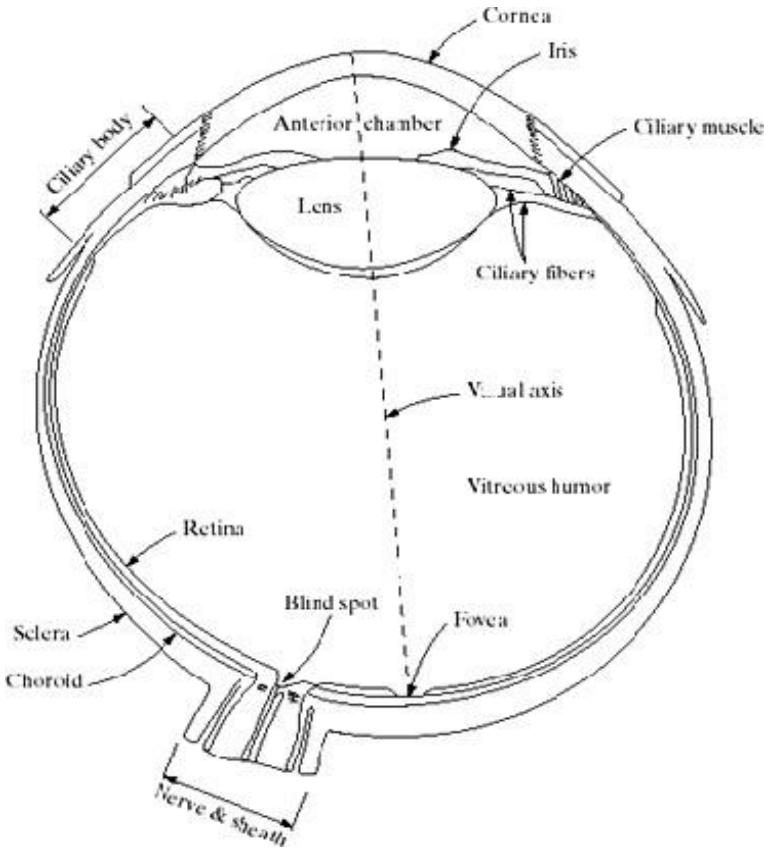
DIGITAL IMAGE FUNDAMENTALS:

ELEMENTS OF VISUAL PERCEPTION:

Although the field of digital image processing is built on a foundation of mathematical and probabilistic formulations, human intuition and analysis play a central role in the choice of one technique versus another, and this choice often is made based on subjective, visual judgments.

Structure of the Human Eye:

Figure shows a simplified horizontal cross section of the human eye. The eye is nearly a sphere, with an average diameter of approximately 20 mm.



Three membranes enclose the eye: the ***cornea*** and ***sclera*** outer cover; the ***choroid*****;** and the ***retina*****.** The cornea is a tough, transparent tissue that covers the anterior surface of the eye. Continuous with the cornea, the sclera is an opaque membrane that encloses the remainder of the optic globe. The choroid lies directly below the sclera.

This membrane contains a network of blood vessels that serve as the major source of nutrition to the eye. Even superficial injury to the choroid, often not deemed serious, can lead to severe eye damage as a result of inflammation that restricts blood flow. The choroid coat is heavily pigmented and hence helps to reduce the amount of extraneous light entering the eye and the backscatter within the optic globe.

At its anterior extreme, the choroid is divided into the ***ciliary*** *body* and the *iris*. The latter contracts or expands to control the amount of light that enters the eye. The central opening of the iris (the pupil) varies in diameter from approximately 2 to 8 mm. The front of the iris contains the visible pigment of the eye, whereas the back contains a black pigment. The ***lens*** is made up of concentric layers of fibrous cells and is suspended by fibers that attach to the ciliary body. It contains 60 to 70% water, about 6% fat, and more protein than any other tissue in the eye.

The lens is colored by a slightly yellow pigmentation that increases with age. In extreme cases, excessive clouding of the lens, caused by the affliction commonly referred to as *cataracts*, can lead to poor

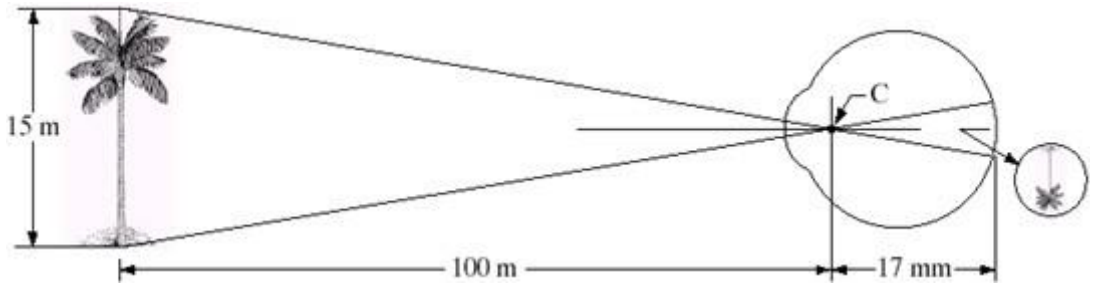
color discrimination and loss of clear vision. The lens absorbs approximately 8% of the visible light spectrum, with relatively higher absorption at shorter wavelengths.

Both infrared and ultraviolet light are absorbed appreciably by proteins within the lens structure and, in excessive amounts, can damage the eye. The innermost membrane of the eye is the ***retina***, which lines the inside of the wall's entire posterior portion. When the eye is properly focused, light from an object outside the eye is imaged on the retina. Pattern vision is afforded by the distribution of discrete light receptors over the surface of the retina.

There are two classes of receptors: ***cones*** and ***rods.*** The cones in each eye number between 6 and 7 million. They are located primarily in the central portion of the retina, called the ***fovea***, and are highly sensitive to color. Humans can resolve fine details with these cones largely because each one is connected to its own nerve end. Muscles controlling the eye rotate the eyeball until the image of an object of interest falls on the fovea. Cone vision is called ***photopic*** or bright-light vision.

Image Formation in the Eye:

In an ordinary photographic camera, the lens has a fixed focal length, and focusing at various distances is achieved by varying the distance between the lens and the imaging plane, where the film (or imaging chip in the case of a digital camera) is located. In the human eye, the converse is true; the distance between the lens and the imaging region (the retina) is fixed, and the focal length needed to achieve proper focus is obtained by varying the shape of the lens. The fibers in the ciliary body accomplish this, flattening or thickening the lens for distant or near objects, respectively.



The distance between the center of the lens and the retina along the visual axis is approximately 17 mm. The range of focal lengths is approximately 14 mm to 17 mm, the latter taking place when the eye is relaxed and focused at distances greater than about 3 m. The geometry in Figure illustrates how to obtain the dimensions of an image formed on the retina.

For example, suppose that a person is looking at a tree 15 m high at a distance of 100 m. Letting *h* denote the height of that object in the retinal image, the geometry of Figure yields 15/100=h/17 or

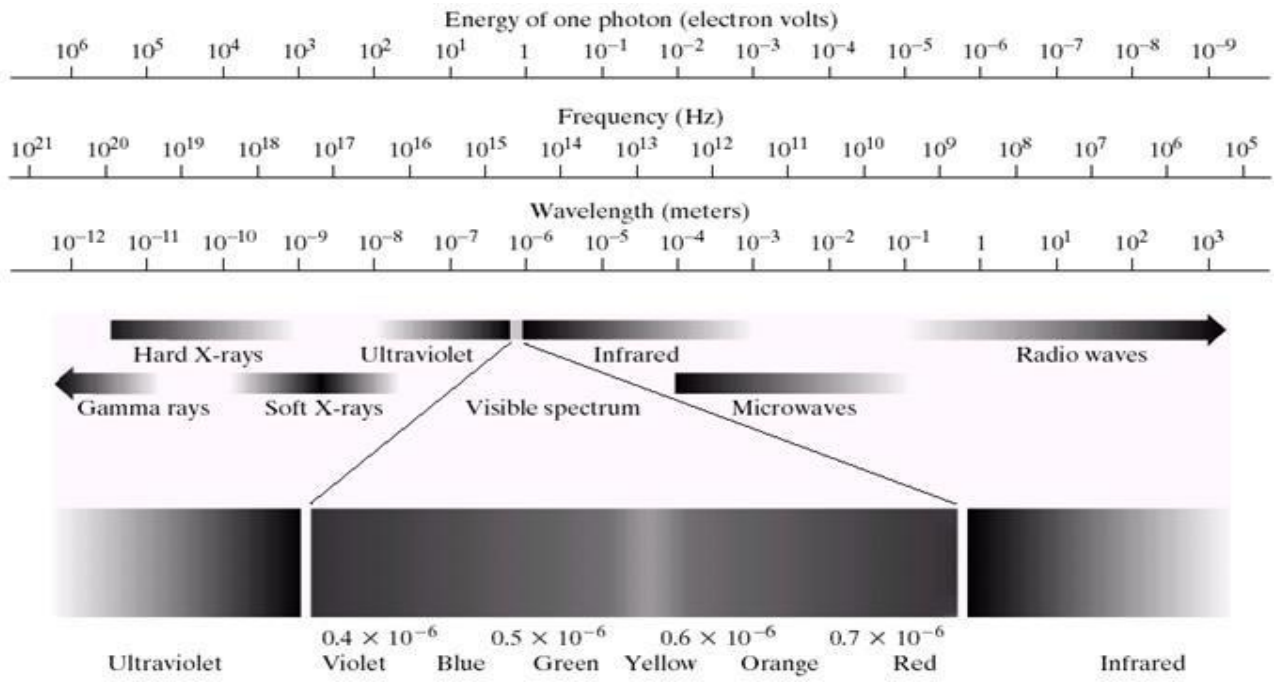
h=2.55mm Perception then takes place by the relative excitation of light receptors, which transform radiant energy into electrical impulses that ultimately are decoded by the brain.

LIGHT AND THE ELECTROMAGNETIC SPECTRUM:

Figure shows, the range of colors we perceive in visible light represents a very small portion of the electromagnetic spectrum. On one end of the spectrum are radio waves with wavelengths billions of times longer than those of visible light. On the other end of the spectrum are gamma rays with wavelengths millions of times smaller than those of visible light.

The electromagnetic spectrum can be expressed in terms of wavelength, frequency, or energy. Wavelength (λ) and frequency (v) are related by the expression

$$\lambda = c/v$$



where *c* is the speed of light($2.998X10^{-8}$m/s). The energy of the various components of the electromagnetic spectrum is given by the expression

$$E = hv$$

where *h* is **Planck's constant**. The units of wavelength are meters, with the terms ***microns*** (denoted and equal to10^{-6}) and ***nanometers*** being used just as frequently. Frequency is measured in **Hertz (Hz)**, with one Hertz being equal to one cycle of a sinusoidal wave per second.

A commonly used unit of energy is the electron-volt. Electromagnetic waves can be visualized as propagating sinusoidal waves with wavelength λ (Fig.a), or they can be thought of as a stream of massless particles, each traveling in a wavelike pattern and moving at the speed of light. Each massless particle contains a certain amount (or bundle) of energy.

Each bundle of energy is called a *photon*. From Equation that energy is proportional to frequency, so the higher-frequency (shorter wavelength) electromagnetic phenomena carry more energy per photon. Thus, radio waves have photons with low energies; microwaves have more energy than radio waves, infrared still more, then visible, ultraviolet, X-rays, and finally gamma rays, the most energetic of all. This is the reason why gamma rays are so dangerous to living organisms.

Light is a particular type of electromagnetic radiation that can be sensed by the human eye. The visible band of the electromagnetic spectrum spans the range from approximately 0.43 μm (violet) to about 0.79 μm (red). For convenience, the color spectrum is divided into six broad regions: violet, blue, green, yellow, orange, and red. No color (or other component of the electromagnetic spectrum) ends abruptly, but rather each range blends smoothly into the next, as shown in the above Figure.

The colors that humans perceive in an object are determined by the nature of the light *reflected* from the object. A body that reflects light relatively balanced in all visible wavelengths appears white to the observer. However, a body that favors reflectance in a limited range of the visible spectrum exhibits some shades of color.

For example, green objects reflect light with wavelengths primarily in the 500 to 570 nm range while absorbing most of the energy at other wavelengths. Light that is void of color is called *monochromatic* (or *achromatic*) *light*.

The only attribute of monochromatic light is its *intensity* or amount. Because the intensity of monochromatic light is perceived to vary from black to grays and finally to white, the term *gray level* is used commonly to denote monochromatic intensity.

The range of measured values of monochromatic light from black to white is usually called the *gray scale*, and monochromatic images are frequently referred to as *gray-scale images*. *Chromatic* (*color*) *light* spans the electromagnetic energy spectrum from approximately 0.43 to 0.79μm as noted previously. In addition to frequency, three basic quantities are used to describe the quality of a chromatic light source:

- Radiance,
- Luminance, and
- Brightness.

Radiance is the total amount of energy that flows from the light source, and it is usually measured in watts (W).

Luminance, measured in lumens (lm), gives a measure of the amount of energy an observer *perceives* from a light source. For example, light emitted from a source operating in the far infrared region of the spectrum could have significant energy (radiance), but an observer would hardly perceive it; its luminance would be almost zero.

Brightness is a subjective descriptor of light perception that is practically impossible to measure. It embodies the achromatic notion of intensity and is one of the key factors in describing color sensation. Gamma radiation is important for medical and astronomical imaging, and for imaging radiation in nuclear environments. Hard (high-energy) X-rays are used in industrial applications. Chest and dental X-rays are in the lower energy (soft) end of the X-ray band. The soft X-ray band transitions into the far ultraviolet light region, which in turn blends with the visible spectrum at longer wavelengths.

IMAGE SENSING AND ACQUISITION:

Most of the images in which we are interested are generated by the combination of an "illumination" source and the reflection or absorption of energy from that source by the elements of the "scene" being imaged.

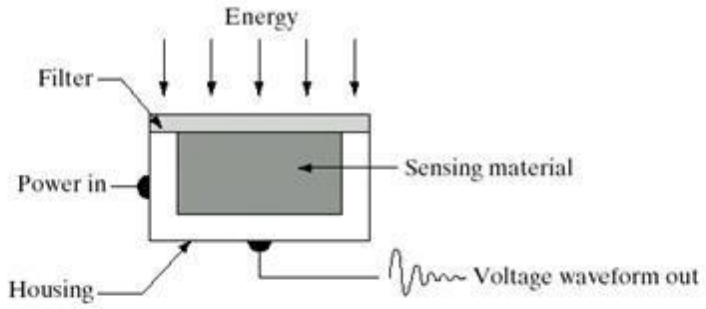
For example, the illumination may originate from a source of electromagnetic energy such as radar, infrared, or X-ray system. An example in the first category is light reflected from a planar surface. An example in the second category is when X-rays pass through a patient's body for the purpose of generating a diagnostic X-ray film.

In some applications, the reflected or transmitted energy is focused onto a photo converter (e.g., a phosphor screen), which converts the energy into visible light. Electron microscopy and some applications of gamma imaging use this approach.

Figure shows the three principal sensor arrangements used to transform
illumination energy into digital images. The idea is simple: Incoming energy is transformed into a voltage by the combination of input electrical power and sensor material that is responsive to the particular type of energy being detected. The output voltage waveform is the response of the sensor(s), and a digital quantity is obtained from each sensor by digitizing its response.

a
b
c

(a) Single imaging sensor.
(b) Line sensor.
(c) Array sensor.



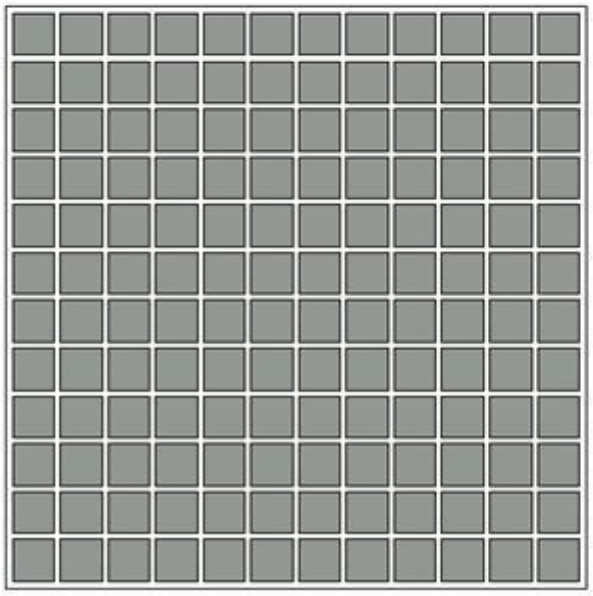
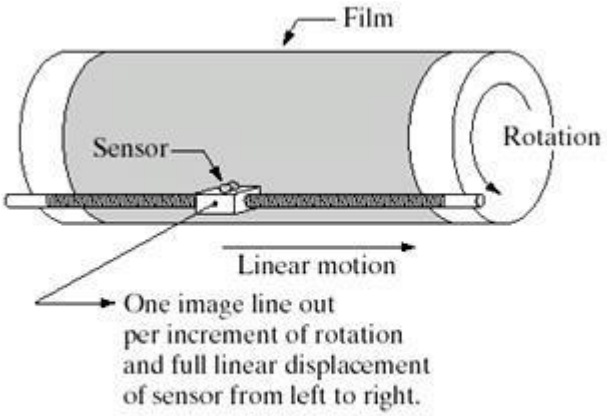


Image Acquisition Using a Single Sensor:

For example, a green (pass) filter in front of a light sensor favors light in the green band of the color spectrum. As a consequence, the sensor output will be stronger for green light than for other components in the visible spectrum. In order to generate a 2-D image using a single sensor, there has to be relative displacements in both the *x*- and *y*-directions between the sensor and the area to be imaged.



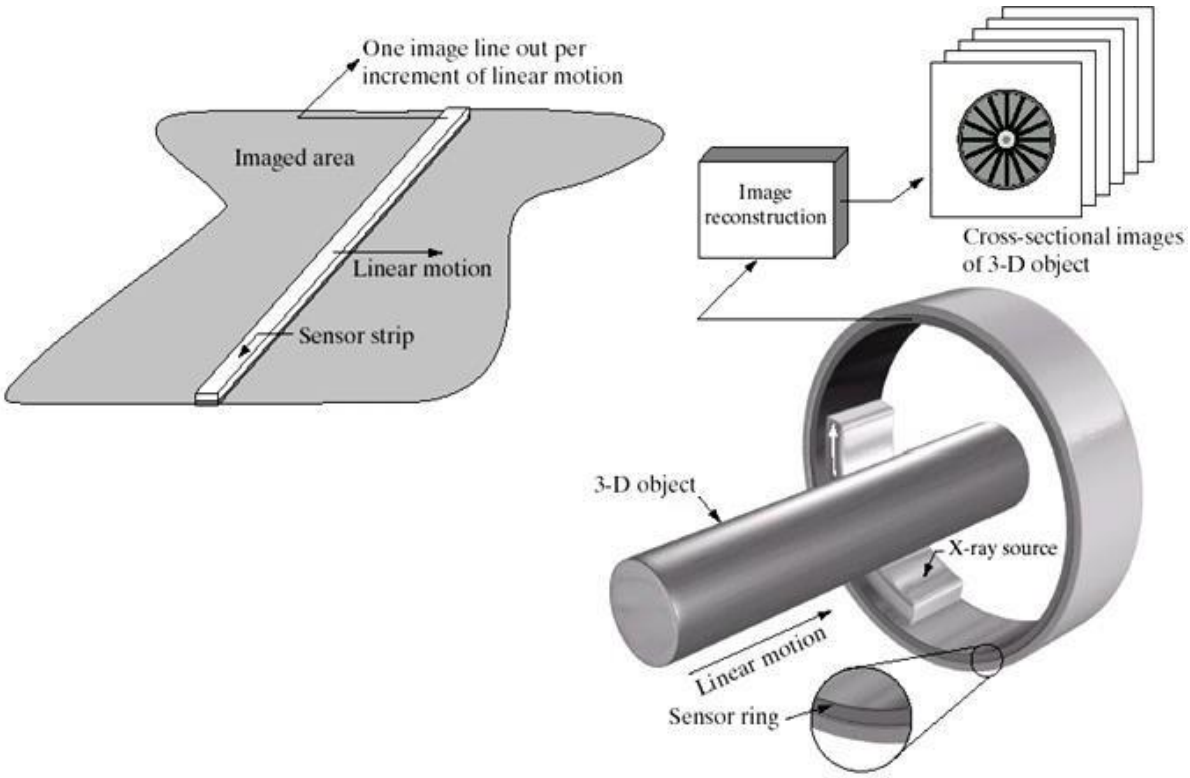
Combining a single sensor with motion to generate a 2-D image.

Figure shows an arrangement used in high-precision scanning, where a film negative is mounted onto a drum whose mechanical rotation provides displacement in one dimension. The single sensor is mounted on a lead screw that provides motion in the perpendicular direction. Because mechanical motion can be controlled with high precision, this method is an inexpensive (but slow) way to obtain high-resolution images.

Other similar mechanical arrangements use a flat bed, with the sensor moving in two linear directions. These types of mechanical digitizers sometimes are referred to as ***microdensitometers***. Another example of imaging with a single sensor places a laser source coincident with the sensor. Moving mirrors are used to control the outgoing beam in a scanning pattern and to direct the reflected laser signal onto the sensor.

Image Acquisition Using Sensor Strips

Motion perpendicular to the strip provides imaging in the other direction, as shown in Fig.(a).This is the type of arrangement used in most flat bed scanners.



a b

(a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

Sensing devices with 4000 or more in-line sensors are possible. In-line sensors are used routinely in airborne imaging applications, in which the imaging system is mounted on an aircraft that flies at a constant altitude and speed over the geographical area to be imaged.

One-dimensional imaging sensor strips that respond to various bands of the electromagnetic spectrum are mounted perpendicular to the direction of flight. The imaging strip gives one line of an image at a time, and the motion of the strip completes the other dimension of a two-dimensional image.

Lenses or other focusing schemes are used to project the area to be scanned onto the sensors. Sensor strips mounted in a ring configuration are used in medical and industrial imaging to obtain cross-sectional ("slice") images of 3-D objects, as
Fig. (b) shows.

A rotating X-ray source provides illumination and the sensors opposite the source collect the X-ray energy that passes through the object (the sensors obviously have to be sensitive to X-ray energy). This is the basis for medical and industrial computerized axial tomography (CAT).

It is important to note that the output of the sensors must be processed by reconstruction algorithms whose objective is to transform the sensed data into meaningful cross-sectional images.

In other words, images are not obtained directly from the sensors by motion alone; they require extensive processing. A 3-D digital volume consisting of stacked images is generated as the object is moved in a direction perpendicular to the sensor ring. Other modalities of imaging based on the CAT principle include magnetic resonance imaging (MRI) and positron emission tomography (PET).The illumination sources, sensors, and types of images are different, but conceptually they are very similar to the basic imaging approach shown in Fig.(b).

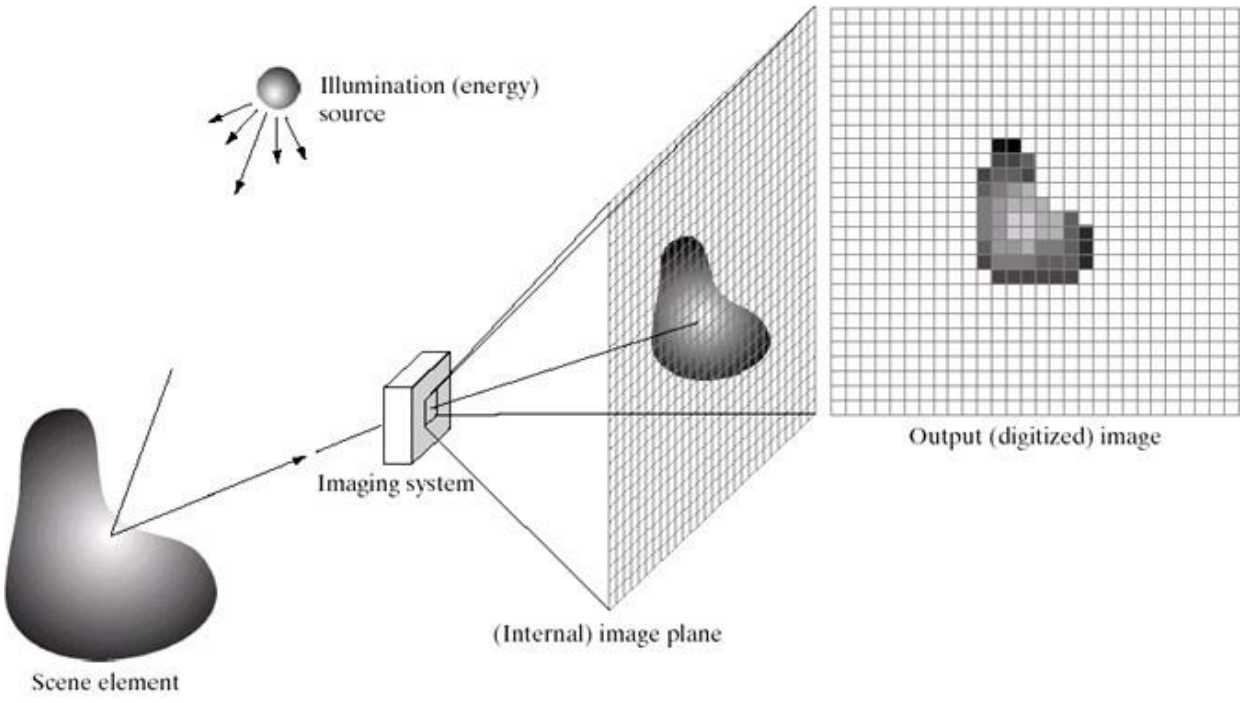
Image Acquisition Using Sensor Arrays

Numerous electromagnetic and some ultrasonic sensing devices frequently are arranged in an array format. This is also the predominant arrangement found in digital cameras. A typical sensor for these cameras is a CCD array, which can be manufactured with a broad range of sensing properties and can be packaged in rugged arrays of elements or more.

CCD sensors are used widely in digital cameras and other light sensing instruments. The response of each sensor is proportional to the integral of the light energy projected onto the surface of the sensor, a property that is used in astronomical and other applications requiring low noise images. Noise reduction is achieved by letting the sensor integrate the input light signal over minutes or even hours. Because the array sensor is two-dimensional, its key advantage is that a complete image can be obtained by focusing the energy pattern onto the surface of the array.

A Simple Image Formation Model

We denote images by two-dimensional functions of the form f(x,y) . The value or amplitude of *f* at spatial coordinates (x,y) is a positive scalar quantity whose physical meaning is determined by the source of the image.



a
b c d e

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

When an image is generated from a physical process, its intensity values are proportional to energy radiated by a physical source (e.g., electromagnetic waves). As a consequence, f(x, y) must be nonzero and finite; that is,

$$0 < f(x, y) < \infty$$

The function f(x,y)may be characterized by two components:

(1) The amount of source illumination incident on the scene being viewed,

(2) The amount of illumination reflected by the objects in the scene.

Appropriately, these are called the *illumination* and *reflectance* components and are denoted by i(x, y) and r(x, y), respectively. The two functions combine as a product to form f(x, y):

$$F(x, y) = i(x, y)\, r(x, y)$$

Where

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

and

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

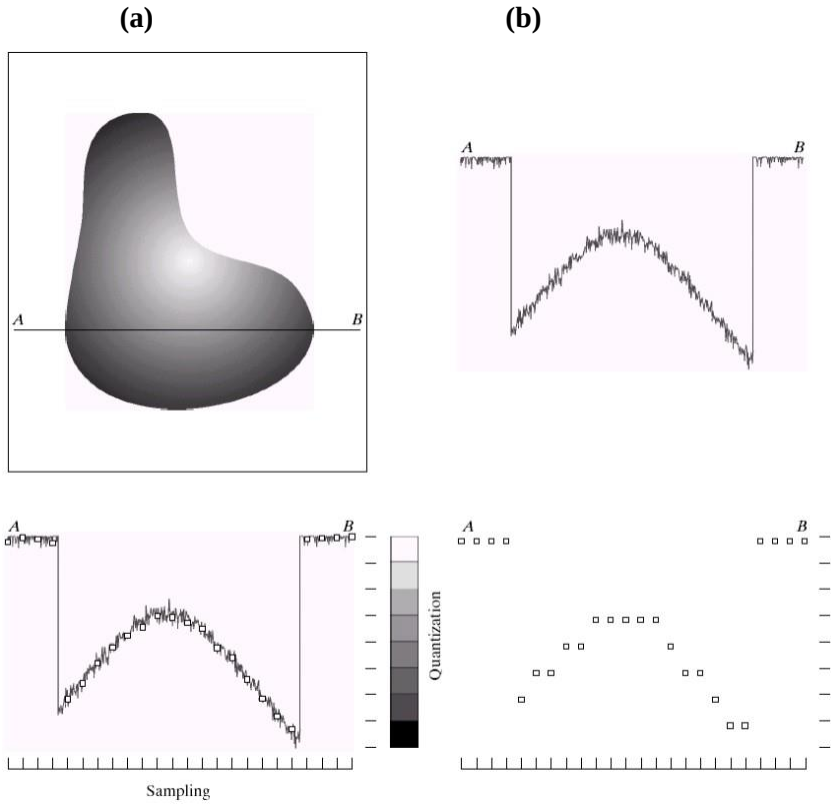
The nature i(x, y) of is determined by the illumination source, and r(x, y) is determined by the characteristics of the imaged objects. It is noted that these expressions also are applicable to images formed via transmission of the illumination through a medium, such as a chest X-ray.

IMAGE SAMPLING AND QUANTIZATION

We see that there are numerous ways to acquire images, but our objective in all is the same: to generate digital images from sensed data. The output of most sensors is a continuous voltage waveform whose amplitude and spatial behavior are related to the physical phenomenon being sensed. To create a digital image, we need to convert the continuous sensed data into digital form. This involves two processes: ***sampling*** and ***quantization*.**

Basic Concepts in Sampling and Quantization

The basic idea behind sampling and quantization is illustrated in Figure. Figure (a) shows a continuous image *f* that we want to convert to digital form. An image may be continuous with respect to the *x*- and *y*-coordinates, and also in amplitude.



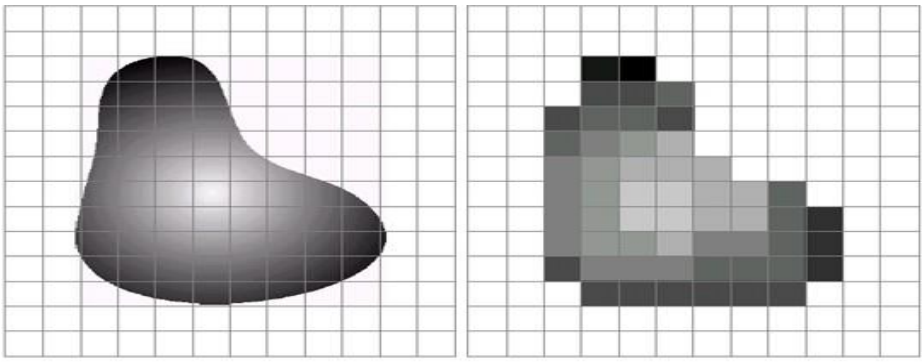
To convert it to digital form, we have to sample the function in both coordinates and in amplitude. Digitizing the coordinate values is called *sampling*. Digitizing the amplitude values is called *quantization*.

The one-dimensional function in Fig.(b) is a plot of amplitude (intensity level) values of the continuous image along the line segment *AB* in Fig.(a).

The random variations are due to image noise. To sample this function; we take equally spaced samples along line *AB*, as shown in Fig.(c).The spatial location of each sample is indicated by a vertical tick mark in the bottom part of the figure. The samples are shown as small white squares superimposed on the function. The set of these discrete locations gives the sampled function. However, the values of the samples still span (vertically) a continuous range of intensity values. In order to form a digital function, the intensity values also must be converted (*quantized*) into discrete quantities.

The right side of Fig. (c) Shows the intensity scale divided into eight discrete intervals, ranging from black to white. The vertical tick marks indicate the specific value assigned to each of the eight intensity intervals. The continuous intensity levels are quantized by assigning one of the eight values to each sample. The assignment is made depending on the vertical proximity of a sample to a vertical tick mark.

The digital samples resulting from both sampling and quantization are shown in Fig. (d). Starting at the top of the image and carrying out this procedure line by line produces a two-dimensional digital image. It is implied in Figure that, in addition to the number of discrete levels used; the accuracy achieved in quantization is highly dependent on the noise content of the sampled signal. Sampling in the manner just described assumes that we have a continuous image in both coordinate directions as well as in amplitude.



a b

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

When a sensing array is used for image acquisition, there is no motion and the number of sensors in the array establishes the limits of sampling in both directions. Quantization of the sensor outputs is as before.

Figure (a) shows a continuous image projected onto the plane of an array sensor. Figure (b) shows the image after sampling and quantization. Clearly, the quality of a digital image is determined to a large degree by the number of samples and discrete intensity levels used in sampling and quantization.

Representing Digital Images:

Let f(s, t) represent a continuous image function of two continuous variables, *s* and *t*. We convert this function into a *digital image* by sampling and quantization, as explained in the previous section. Suppose that we sample the continuous image into a 2-D array, f(x, y), containing *M* rows and *N* columns, where (x, y) are discrete coordinates.

For notational clarity and convenience, we use integer values for these discrete coordinates: x = 0, 1, 2,……..,M-1 and y = 0, 1,2,… ,N-1 Thus, for example, the value of the digital image at the origin is F(0,0) , and the next coordinate value along the first row is f(0,1) . Here, the notation (0, 1) is used to signify the second sample along the first row.

It *does not* mean that these are the values of the physical coordinates when the image was sampled. In general, the value of the image at any coordinates (x, y) is denoted f(x, y), where *x* and *y* are integers. The section of the real plane spanned by the coordinates of an image is called the *spatial domain*, with *x* and *y* being referred to as *spatial variables* or *spatial coordinates*.

Spatial and Intensity Resolution:

Spatial resolution is a measure of the smallest discernible detail in an image. Quantitatively, *spatial resolution* can be stated in a number of ways, with *line pairs per unit distance*, and *dots* (*pixels*) *per unit distance* being among the most common measures. Suppose that we construct a chart with alternating black and white vertical lines, each of width *W* units (*W* can be less than 1).

The width of a *line pair* is thus 2*W*, and there are *W* line pairs per unit distance. For example, if the width of a line is 0.1 mm, there are 5 line pairs per unit distance (mm). A widely used definition of image resolution is the largest number of *discernible* line pairs per unit distance (e.g., 100 line pairs per mm). Dots per unit distance are a measure of image resolution used commonly in the printing and publishing industry.

In the U.S., this measure usually is expressed as *dots per inch* (dpi). To give you an idea of quality, newspapers are printed with a resolution of 75 dpi, magazines at 133 dpi, glossy brochures at 175 dpi, and the book page at which you are presently looking is printed at 2400 dpi. The key point in the preceding paragraph is that, to be meaningful, measures of spatial resolution must be stated with respect to spatial units.

Image size by itself does not tell the complete story. To say that an image has, say, a resolution 1024 X 1024 pixel is not a meaningful statement without stating the spatial dimensions encompassed by the image. Size by itself is helpful only in making comparisons between imaging capabilities. For example, a digital camera with a 20-megapixel CCD imaging chip can be expected to have a higher capability to resolve detail than an 8-megapixel camera, assuming that both cameras are equipped with comparable lenses and the comparison images are taken at the same distance.

Intensity resolution similarly refers to the smallest discernible change in intensity level. We have considerable discretion regarding the number of samples used to generate a digital image, but this is not true regarding the number of intensity levels.

Based on hardware considerations, the number of intensity levels usually is an integer power of two, as mentioned in the previous section. The most common number is 8 bits, with 16 bits being used in some applications in which enhancement of specific intensity ranges is necessary.

Image Interpolation:

Interpolation is a basic tool used extensively in tasks such as zooming, shrinking, rotating, and geometric corrections. Our principal objective in this section is to introduce interpolation and apply it to image resizing (shrinking and zooming), which are basically image *resampling* methods. Uses of interpolation in applications such as rotation and geometric corrections.

Fundamentally, *interpolation* is the process of using known data to estimate values at unknown locations. We begin the discussion of this topic with a simple example. Suppose that an image of size 500 X 500 pixels has to be enlarged 1.5 times to 750 X 750 pixels.

A simple way to visualize zooming is to create an imaginary 750 X 750 grid with the same pixel spacing as the original, and then shrink it so that it fits exactly over the original image. Obviously, the pixel spacing in the shrunken 750 X 750 grid will be less than the pixel spacing in the original image.

To perform intensity-level assignment for any point in the overlay, we look for its closest pixel in the 750 X 750 original images and assign the intensity of that pixel to the new pixel in the grid. When we are finished assigning intensities to all the points in the overlay grid, we expand it to the original specified size to obtain the zoomed image.

SOME BASIC RELATIONSHIPS BETWEEN PIXELS

In this section, consider several important relationships between pixels in a digital image. As mentioned before, an image is denoted by f(x, y).When referring in this section to a particular pixel, we use lowercase letters, such as *p* and *q*.

Neighbors of a Pixel:

A pixel *p* at coordinates f(x, y) has four *horizontal* and *vertical* neighbors whose coordinates are given by

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

This set of pixels, called the 4-*neighbors* of *p*, is denoted by $N_4(P)$ Each pixel is a unit distance from (x, y), and some of the neighbor locations of *p* lie outside the digital image if (x, y) is on the border of the image. The four *diagonal* neighbors of *p* have coordinates and are denoted by $N_D(P)$.

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

These points, together with the 4-neighbors, are called the 8-*neighbors* of *p*, denoted by $N_4(P)$. As before, some of the neighbor locations in $N_D(P)$ and $N_8(P)$ fall outside the image if (x, y)is on the border of the image.

Adjacency, Connectivity, Regions, and Boundaries:

Let *V* be the set of intensity values used to define adjacency. In a binary image V= {1}, if we are referring to adjacency of pixels with value 1. In a gray-scale image, the idea is the same, but set *V* typically contains more elements. For example, in the adjacency of pixels with a range of possible intensity values 0 to 255, set *V* could be any subset of these 256 values. We consider three types of adjacency:

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

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

(c) *m-adjacency* (mixed adjacency).Two pixels *p* and *q* with values from *V* are

djacent if

(i) *q* is in $N_4(P)$ *or*

(ii) *q* is in $N_D(P)$ *and* the set $N_4(P) \cap N_4(q)$ has no pixels whose values are from *V*.

Mixed adjacency is a modification of 8-adjacency. It is introduced to eliminate the ambiguities that often arise when 8-adjacency is used.

Distance Measures

For pixels *p*, *q*, and *z*, with coordinates (*x*, *y*), (*s*, *t*), and (v, w), respectively, *D* is a *distance function* or *metric* if

- ***D(p, q)≥0 (D(p, q)=0 if p=q),***
- ***D(p, q)= D(q, p), and***
- ***D(p, z)≤ D(p, q)+ D(q, z)***

The *Euclidean distance* between *p* and *q* is defined as

$$D_e(p, q) = [(x-s)^2+(y-t)^2]^{1/2}$$

For this distance measure, the pixels having a distance less than or equal to some value *r* from (*x*, *y*) are the points contained in a disk of radius *r* centered at (*x*, *y*).

The D_4 *distance* (called the *city-block distance*) between *p* and *q* is defined as

$$D_4(p, q) = |x - s| + |y - t|$$

In this case, the pixels having a D_4 distance from (*x*, *y*) less than or equal to some value *r* form a diamond centered at (*x*, *y*). For example, the pixels with D_4 distance ≤ 2 from (*x*, *y*) (the center point) form the following contours of constant distance:

2

2 1 2

2 1 0 1 2

2 1 2

2

The pixels with $D_4 = 1$ are the 4-neighbors of (x, y). The D_8 distance (called the *chessboard distance*) between p and q is defined as

$$D_8(p, q) = \max(|x - s|, |y - t|)$$

In this case, the pixels with distance from (x, y) less than or equal to some value r form a square centered at. For example, the pixels with D_8 **distance** ≤ 2 from (x, y) (the center point) form the following contours of constant distance:

2 2 2 2 2

2 1 1 1 2

2 1 0 1 2

2 1 1 1 2

2 2 2 2 2

The pixels with $D_8 = 1$ are the 8-neighbors of (x, y).

QUESTION BANK

5 MARKS:

1. Explain Types of computerized processes in detail.
2. Explain Components of an image processing system in detail **(April 2012)**.
3. Explain Light and the electromagnetic spectrum in detail.
4. Explain some basic relationships between pixels in detail **(April 2012)**.

10 MARKS:

1. Explain Examples of fields that use digital image processing in detail.
2. Explain Fundamental steps in digital image processing in detail **(April 2012)**.
3. Explain Image sensing and acquisition in detail **(April 2012)**..
4. Explain Image sampling and quantization in detail.

UNIT-II

DATA STRUCTURES IN IMAGE ANALYSIS:

Introduction:

- Computer program = data + algorithm
- Data organization can considerably affect the simplicity of the selection and the implementation of an algorithm. The choice of data structures is fundamental when writing a program

Levels of image data representation:

- **Iconic images** - consists of images containing original data; integer matrices with data about pixel brightness.
- E.g., outputs of pre-processing operations (e.g., filtration or edge sharpening) used for highlighting some aspects of the image important for further treatment.
- **Segmented images** - parts of the image are joined into groups that probably belong to the same objects.
- It is useful to know something about the application domain while doing image segmentation; it is then easier to deal with noise and other problems associated with erroneous image data.
- **Geometric representations** - hold knowledge about 2D and 3D shapes.
- The quantification of a shape is very difficult but very important.
- **Relational models** - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
- A priori knowledge
    - position of the airport (e.g., from a map)
    - relations to other objects in the image (e.g., to roads, lakes, urban areas)
    - Geometric models of planes for which we are searching etc.

Traditional image data structures:

It is not only for the direct representation of image information, also a basis of more complex hierarchical methods of image representation.

- matrices,
- chains,
- graphs,

- lists of object properties,
- relational databases,

Matrices:

- Most common data structure for low level image representation Elements of the matrix are integer numbers
- Image data of this kind are usually the direct output of the image capturing device, e.g., a scanner.

Algorithm 3.1: Co-occurrence matrix $C_r(z, y)$ for the relation r

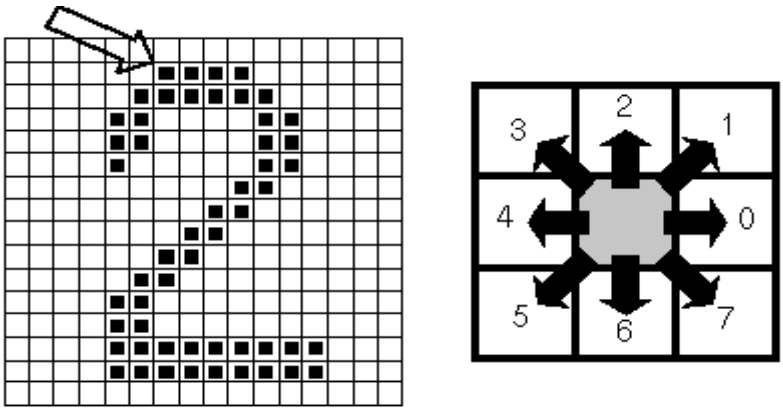
1. Assign $C_r(z, y) = 0$ for all z, $y \in [0, L]$, where L is the maximum brightness.

2. For all pixels (i_1, j_1) in the image, determine (i_2, j_2) which has the relation r with the pixel (i_1, j_1), and perform

$$C_r(f(i_1, j_1), f(i_2, j_2)) = C_r(f(i_1, j_1), f(i_2, j_2)) + 1$$

Chains:

- **Chains** are used for description of object borders
- Symbols in a chain usually correspond to the neighborhood of primitives in the image.



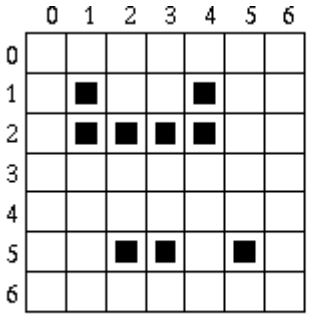
An example chain code; the reference pixel is marked by an arrow:
00007766555555660000000644444444222111111223444565221 1.

- If local information is needed from the chain code, it is necessary to search through the whole chain systematically.
- Does the border turn somewhere to the left by 90 degrees?

- A sample pair of symbols in the chain must be found - simple.
- If global information is needed, situation is much more difficult.
- Questions about the shape of the border represented by chain codes are not trivial.
- Chains can be represented using static data structures (e.g., 1D arrays); their size is the longest length of the chain expected.
- Dynamic data structures are more advantageous to save memory.

Run length coding:

- Often used to represent strings of symbols in an image matrix (e.g., FAX machines use it).
- In binary images, run length coding records only areas that belong to the object in the image; the area is then represented as a list of lists.
- Each row of the image is described by a sublist, the first element of which is the row number.
- Subsequent terms are co-ordinate pairs; the first element of a pair is the beginning of a run andthe second is the end.
- There can be several such sequences in the row.



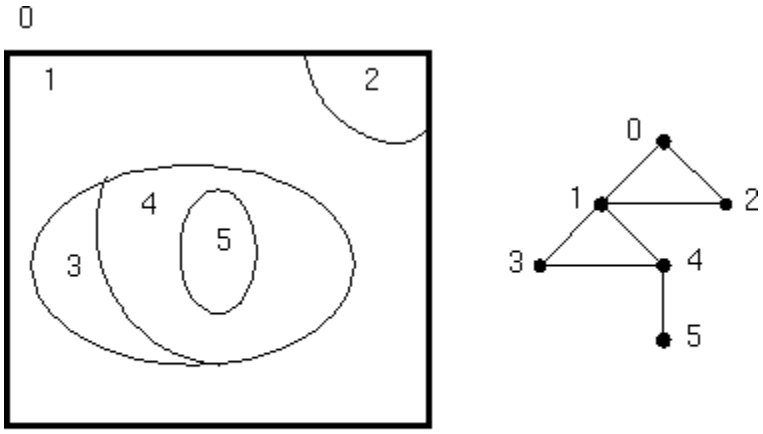
Run length coding; the code is ((11144)(214)(52355)).

- Run length coding can be used for an image with multiple brightness levels as well - in the sublist, sequence brightness must also be recorded.

Topological data structures:

Image description as a set of **elements and their relations**.

- Graphs
- Evaluated graphs
- Region adjacency graphs



An example region adjacency graph.

Relational structures:

- information is then concentrated in relations between semantically important parts of the image - objects - that are the result of segmentation
- Appropriate for higher level image understanding

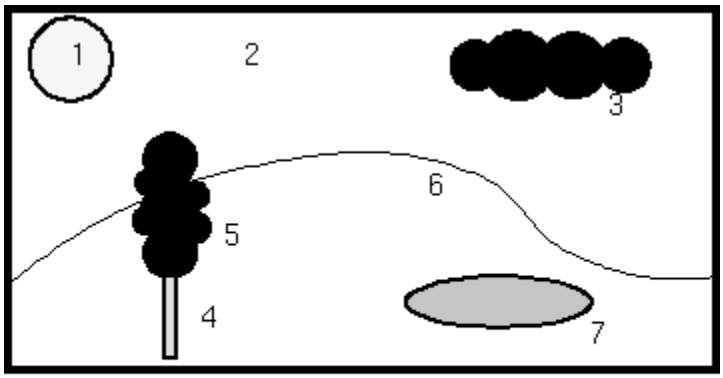


Figure 3.4 Description of objects using relational structure.

No.	Object name	Colour	Min. row	Min. col.	Inside
1	sun	white	5	40	2
2	sky	blue	0	0	-
3	cloud	grey	20	180	2
4	tree trunk	brown	95	75	6
5	tree crown	green	53	63	-
6	hill	light green	97	0	-
7	pond	blue	100	160	6

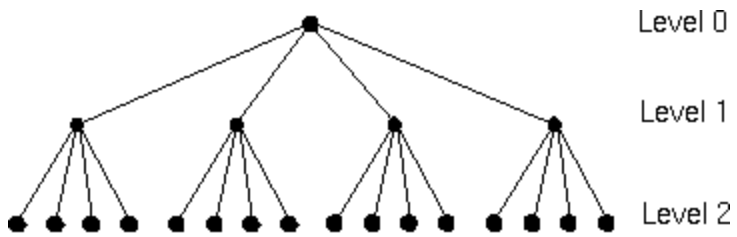
Table 3.1 *Relational table*

Hierarchical data structures:

- Computer vision is by its nature very computationally expensive, if for no other reason than the amount of data to be processed.
- One of the solutions is using parallel computers = brute force
- Many computer vision problems are difficult to divide among processors, or decompose in any way.
- Hierarchical data structures make it possible to use algorithms which decide a strategy for processing on the basis of relatively small quantities of data.
- They work at the finest resolution only with those parts of the image for which it is necessary, using knowledge instead of brute force to ease and speed up the processing.
- Two typical structures –
    - **Pyramids**
    - **Quadtrees**

Pyramids:

- **M-pyramid** - Matrix pyramid ... is a **sequence** $\{M_L, M_{L-1}, ..., M_0\}$ of images
- M_L has the same dimensions and elements as the original image
- M_{i-1} is derived from the M_i by reducing the resolution by one half.
- Square matrices with dimensions equal to powers of two required - M_0 corresponds to one pixel only.
- M-pyramids are used when it is necessary to work with an image at different resolutions simultaneously.
- An image having one degree smaller resolution in a pyramid contains four times less data, so that it is processed approximately four times as quickly.
- Often it is advantageous to use several resolutions simultaneously rather than to choose just one image from the M-pyramid.
- Such images can be represented using tree pyramids ... **T-pyramids**.
- T-pyramid is a **tree**, every node of the T-pyramid has 4 child nodes.



T-pyramid.

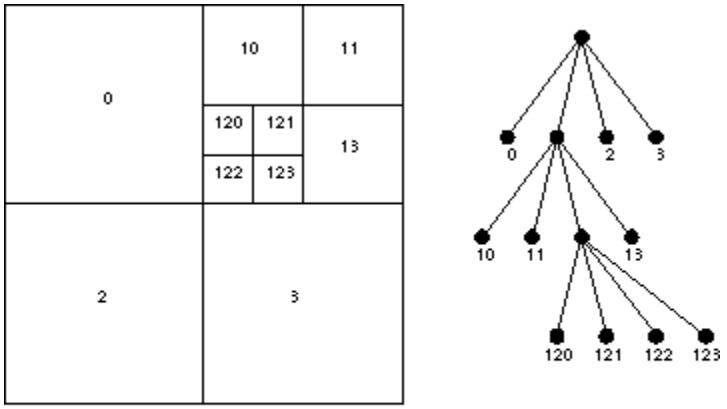
- The number of image pixels used by an M-pyramid for storing all matrices is

$$N^2 \left(1 + \frac{1}{4} + \frac{1}{16} + \ldots\right) \approx 1.33\, N^2$$

- The T-pyramid is represented in memory similarly.
- Arcs of the tree need not be recorded because addresses of the both child and parent nodes are easy to compute due to the regularity of the structure. An algorithm for the effective creation

Quadtrees:

- Quadtrees are modifications of T-pyramids.
- Every node of the tree except the leaves has four children (NW: north-western, NE: north-eastern, SW: south-western, SE: south-eastern).
- Similarly to T-pyramids, the image is divided into four quadrants at each hierarchical level; however it is not necessary to keep nodes at all levels.
- If a parent node has four children of the same value (e.g., brightness), it is not necessary to record them.



Quadtree.

- Problems associated with hierarchical image representation:
    - Dependence on the position, orientation and relative size of objects.
    - Two similar images with just very small differences can have very different pyramid or quadtree representations.
    - Even two images depicting the same, slightly shifted scene, can haveentirely different representations.

IMAGE PRE-PROCESSING:

Pixel brightness transformations:

- Pre-processing is a common name for operations with images at the lowest level of abstraction -- both input and output are intensity images.
- The aim of pre-processing is an improvement of the image data that suppresses unwanted distortions or enhances some image features important for further processing.
- Four categories of image pre-processing methods according to the size of the pixel neighborhood that is used for the calculation of a new pixel brightness:
    - pixel brightness transformations,
    - geometric transformations,
    - pre-processing methods that use a local neighborhood of the processed pixel, and
    - Image restoration that requires knowledge about the entire image.
- Other classifications of image pre-processing methods exist.
- Image pre-processing methods use the considerable redundancy in images.
- Neighboring pixels corresponding to one object in real images have essentially the same or similar brightness value.
- Thus, distorted pixel can often be restored as an average value of neighboring pixels.
- If pre-processing aims to correct some degradation in the image, the nature of a priori information is important:
    - Knowledge about the nature of the degradation; only very general properties of the degradation are assumed.
    - Knowledge about the properties of the image acquisition device, and conditions under which the image was obtained. The nature of noise (usually its spectral characteristics) is sometimes known.
    - Knowledge about objects that are searched for in the image, which may simplify the pre-processing very considerably.
- If knowledge about objects is not available in advance it can be estimated during the processing.

Pixel brightness transformations:

- Brightness transformations modify pixel brightness -- the transformation depends on the properties of a pixel itself.
    - ❖ Brightness corrections
    - ❖ Gray scale transformations

Brightness correction:

- Considers original brightness
- Pixel position in the image.

Gray scale transformations:

- Change brightness without regard to position in the image.

Position dependent brightness correction

- Ideally, the sensitivity of image acquisition and digitization devices should not depend on position in the image, but this assumption is not valid in many practical cases.
- Sources of degradation.
    - ❖ Uneven sensitivity of light sensors
    - ❖ Uneven object illumination
- Systematic degradation can be suppressed by brightness correction.
- Let a multiplicative error coefficient e(i,j) describe the change from the ideal identity transfer function;
    - g(i,j) is the original undegraded image (or desired image)
    - f(i,j) is the image containing degradation.

$$f(i,j) = e(i,j)\, g(i,j)$$

- If a reference image g(i,j) is known (e.g., constant brightness c) then
    - the degraded result is $f_c(i,j)$
    - systematic brightness errors can be suppressed:

$$g(i,j) = \frac{f(i,j)}{e(i,j)} = \frac{c\, f(i,j)}{f_c(i,j)}$$

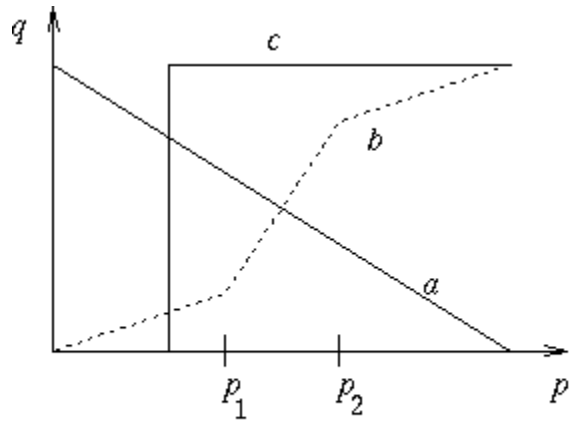
- Image degradation process must be stable,
    - the device should be calibrated time to time (find error coefficients e(i,j))

- This method implicitly assumes linearity of the transformation, which is not true in reality as the brightness scale is limited into some interval.
    - overflow is possible
        - The best reference image has brightness that is far enough from both limits.
- If the gray scale has 256 brightness's the ideal image has constant brightness value 128.
    - Most TV cameras have automatic control of the gain which allows them to operate under changing illumination conditions. If systematic errors are suppressed using error coefficients, this automatic gain control should be switched off first.

Gray scale transformation:

- Grey scale transformations do not depend on the position of the pixel in the image.
- Brightness transform:

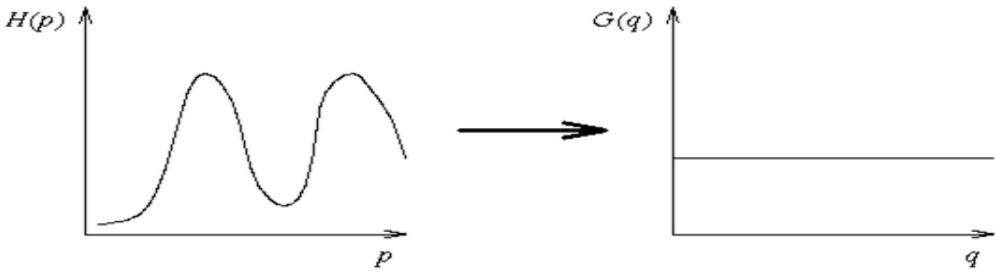
$$q = T(p)$$



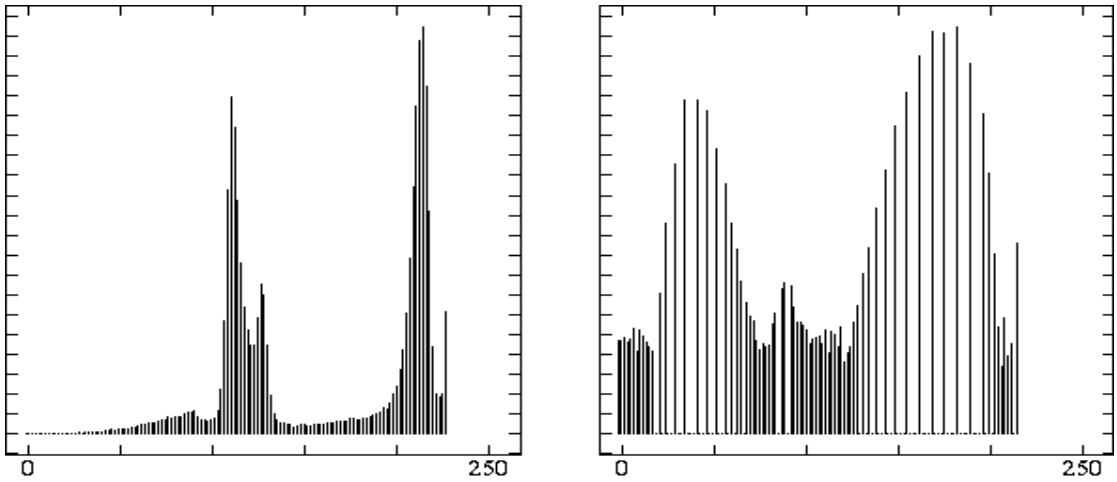
Some grey scale transformations.

a - Negative transformation
b - contrast enhancement (between p1 and p2)
c - Brightness thresholding

- Grey scale transformations can be performed using look-up tables.
- Grey scale transformations are mostly used if the result is viewed by a human.
- Typical grey level transform ... histogram equalization is usually found automatically.
- The aim - image with equally distributed brightness levels over the whole brightness scale



Histogram equalization.



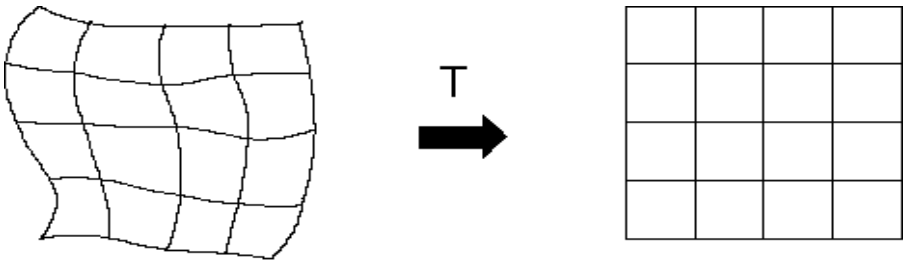
Histogram equalization: Original and equalized histograms.

- The histogram can be treated as a discrete probability density function.
- The monotonic property of the transform T implies.

$$\sum_{i=0}^{k} G(q_i) = \sum_{i=0}^{k} H(p_i)$$

GEOMETRIC TRANSFORMATIONS:

- Geometric transforms permit the elimination of geometric distortion that occurs when an image is captured.
- An example is an attempt to match remotely sensed images of the same area taken after one year, when the more recent image was probably not taken from precisely the same position.
- To inspect changes over the year, it is necessary first to execute a geometric transformation, and then subtract one image from the other.



Geometric transform on a plane.

- A geometric transform is a vector function T that maps the pixel (x,y) to a new position (x',y').

$$x' = T_x(x, y), \qquad y' = T_y(x, y)$$

- The transformation equations are either known in advance or can be determined from known original and transformed images.
- Several pixels in both images with known correspondence are used to derive the unknown transformation.
- A geometric transform consists of two basic steps ...

  1. Determining the pixel co-ordinate transformation
     - Mapping of the co-ordinates of the input image pixel to the point in the output image.
     - The output point co-ordinates should be computed as continuous values (real numbers) as the position does not necessarily match the digital grid after the transform.

  2. Finding the point in the digital raster which matches the transformed point and determining its brightness.
- Brightness is usually computed as an interpolation of the brightnesses of several points in the neighborhood.

Pixel co-ordinate transformations:

- General case of finding the co-ordinates of a point in the output image after a geometric transform.
  - usually approximated by a polynomial equation

$$x' = \sum_{r=0}^{m} \sum_{k=0}^{m-r} a_{rk}\, x^r\, y^k \qquad y' = \sum_{r=0}^{m} \sum_{k=0}^{m-r} b_{rk}\, x^r\, y^k$$

- This transform is linear with respect to the coefficients a_{rk}, b_{rk}
- If pairs of corresponding points (x,y), (x',y') in both images are known, it is possible to determine a_{rk}, b_{rk} by solving a set of linear equations.
- More points than coefficients are usually used to get robustness.

- If the geometric transform does not change rapidly depending on position in the image, low order approximating polynomials, m=2 or m=3, are used, needing at least 6 or 10 pairs of corresponding points.
- The corresponding points should be distributed in the image in a way that can express the geometric transformation - usually they are spread uniformly.
- The higher the degree of the approximating polynomial, the more sensitive to the distribution of the pairs of corresponding points the geometric transform.
- In practice, the geometric transform is often approximated by the **bilinear transformation**

  - 4 pairs of corresponding points are sufficient to find transformation coefficients

$$x' = a_0 + a_1 x + a_2 y + a_3 xy$$
$$y' = b_0 + b_1 x + b_2 y + b_3 xy$$

- Even simpler is the **affine transformation** for which three pairs of corresponding points are sufficient to find the coefficients

$$x' = a_0 + a_1 x + a_2 y$$
$$y' = b_0 + b_1 x + b_2 y$$

- The affine transformation includes typical geometric transformations such as
  - rotation,
  - translation,
  - scaling
  - Skewing.
- A geometric transform applied to the whole image may change the co-ordinate system, and a Jacobean J provides information about how the co-ordinate system changes.

$$J = \left| \frac{\partial(x', y')}{\partial(x, y)} \right| = \begin{vmatrix} \frac{\partial x'}{\partial x} & \frac{\partial x'}{\partial y} \\ \frac{\partial y'}{\partial x} & \frac{\partial y'}{\partial y} \end{vmatrix}$$

- If the transformation is singular (has no inverse) then J=0. If the area of the image is invariant under the transformation then J=1.
- The Jacobean for the general bilinear transform (4.11)

$$J = a_1 b_2 - a_2 b_1 + (a_1 b_3 - a_3 b_1)x + (a_3 b_2 - a_2 b_3)y \qquad (4.14)$$

- The Jacobean for the affine transformation (4.12)

$$J = a_1 b_2 - a_2 b_1 \qquad (4.15)$$

- Important geometric transformations:
- **Rotation** - by the angle phi about the origin

$$\begin{aligned} x' &= x \cos\phi + y \sin\phi \\ y' &= -x \sin\phi + y \cos\phi \\ J &= 1 \end{aligned} \qquad (4.16)$$

- **Change of scale** - a in the x axis and b in the y axis

$$\begin{aligned} x' &= ax \\ y' &= bx \\ J &= ab \end{aligned} \qquad (4.17)$$

- **Skewing** by the angle phi

$$\begin{aligned} x' &= x + y \tan\phi \\ y' &= y \\ J &= 1 \end{aligned} \qquad (4.18)$$

- Complex geometric transformations (disortion)
  - ✓ approximation by partitioning an image into smaller rectangular subimages;
  - ✓ for each subimage, a simple geometric transformation, such as the affine, is estimated using pairs of corresponding pixels.
  - ✓ geometric transformation (distortion) is then performed separately in each subimage.
- Typical geometric distortions which have to be overcome in remote sensing:
  - o distortion of the optical systems
  - o nonlinearities in row by row scanning
  - o nonconstant sampling period.

Brightness interpolation:

- Assume that the planar transformation has been accomplished, and new point co-ordinates (x',y') were obtained.
- The position of the point does not in general fit the discrete raster of the output image.
- Values on the integer grid are needed.
- Each pixel value in the output image raster can be obtained by **brightness interpolation** of some neighboring non integer samples.
- The brightness interpolation problem is usually expressed in a dual way (by determining the brightness of the original point in the input image that corresponds to the point in the output image lying on the discrete raster).
- Computing the brightness value of the pixel (x',y') in the output image where x' and y' lie on the discrete raster

$$(x, y) = \mathrm{T}^{-1}(x', y') \qquad (4.19)$$

- In general the real co-ordinates after inverse transformation (dashed lines in Figures) do not fit the input image discrete raster (solid lines), and so brightness is not known.
- To get the brightness value of the point (x,y) the input image is resampled.

$$f_n(x, y) = \sum_{l=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} g_s(l\Delta x,\ k\Delta y)\, h_n(x - l\Delta x,\ y - k\Delta y) \qquad (4.20)$$

- f_{n}(x,y) ... result of interpolation
- h_{n} is the **interpolation kernel**
  - ✓ Usually, a small neighborhood is used, outside which h_{n} is zero.

Nearest neighbor interpolation:

- assigns to the point (x,y) the brightness value of the nearest point g in the discrete raster

$$f_1(x, y) = g_s(round(x),\ round(y)) \qquad (4.21)$$

- The right side of Figure shows how the new brightness is assigned.
- Dashed lines show how the inverse planar transformation maps the raster of the output image into the input image - full lines show the raster of the input image.

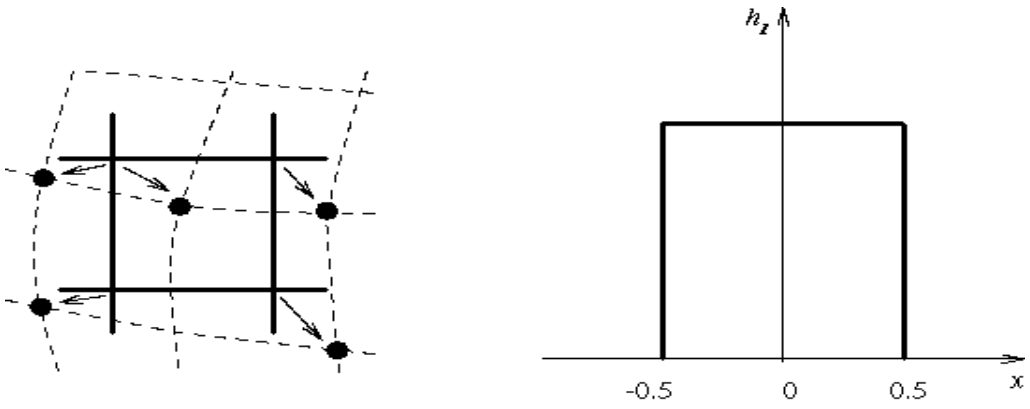


Figure 4.7 Nearest neighbourhood interpolation.

- The position error of the nearest neighborhood interpolation is at most half a pixel.
- This error is perceptible on objects with straight line boundaries that may appear step-like after the transformation.

Linear interpolation:

- Explores four points neighboring the point (x,y), and assumes that the brightness function islinear in this neighborhood.

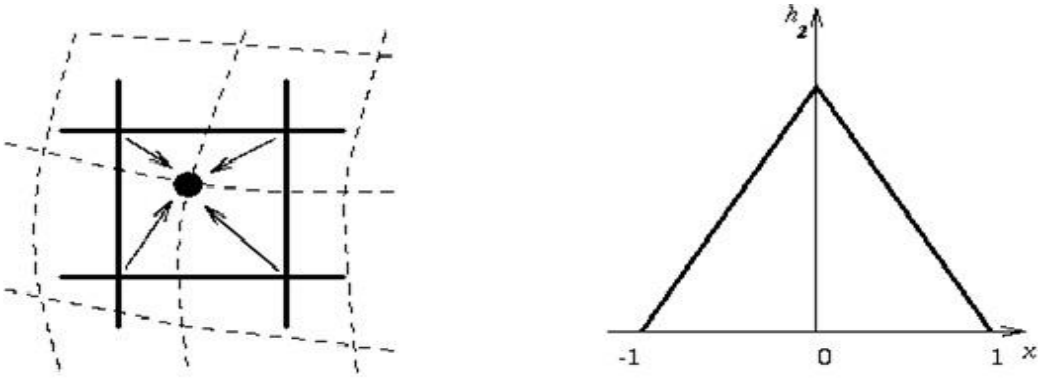


Figure 4.8 Linear interpolation.

- Linear interpolation is given by the equation

$$
\begin{aligned}
f_2(x,y) = \; & (1-a)(1-b)\, g_s(l,k) \\
& +a(1-b)\, g_s(l+1,k) \\
& +b(1-a)\, g_s(l,k+1) \\
& +ab\, g_s(l+1,k+1)
\end{aligned} \qquad (4.22)
$$

$$
\begin{aligned}
l &= floor\,(x), \quad & a &= x-l \\
k &= floor\,(y), \quad & b &= y-k
\end{aligned}
$$

- Linear interpolation can cause a small decrease in resolution and blurring due to its averaging nature.
- The problem of step like straight boundaries with the nearest neighborhood interpolation is reduced.

Bicubic Interpolation:

- Improves the model of the brightness function by approximating it locally by a bicubic polynomial surface; sixteen neighboring points are used for interpolation.
- Interpolation kernel (`Mexican hat') is given by

$$
h_3 = \begin{cases} 1-2|x|^2+|x|^3 & \text{for } 0<|x|<1 \\ 4-8|x|+5|x|^2-|x|^3 & \text{for } 1<|x|<2 \\ 0 & \text{otherwise} \end{cases} \qquad (4.23)
$$

- Bicubic interpolation does not suffer from the step-like boundary problem of nearest neighborhood interpolation, and copes with linear interpolation blurring as well.
- Bicubic interpolation is often used in raster displays that enable zooming with respect to an arbitrary point -- if the nearest neighborhood method were used, areas of the same brightness would increase.
- Bicubic interpolation preserves fine details in the image very well.

LOCAL PRE-PROCESSING:

- Pre-processing methods use a small neighborhood of a pixel in an input image to get a new brightness value in the output image.
- Such pre-processing operations are called also **filtration**.
- Local pre-processing methods can be divided into the two groups according to the goal of the processing:
- **Smoothing** aims to suppress noise or other small fluctuations in the image
    - ✓ equivalent to the suppression of high frequencies in the frequency domain.

- Unfortunately, smoothing also blurs all sharp edges that bear important information about the image.
- **Gradient operators** are based on local derivatives of the image function.
- Derivatives are bigger at locations of the image where the image function undergoes rapid changes. The aim of gradient operators is to indicate such locations in the image.
- Gradient operators have a similar effect as suppressing low frequencies in the frequency domain.
- Noise is often high frequency in nature; unfortunately, if a gradient operator is applied to an image the noise level increases simultaneously.
- Clearly, smoothing and gradient operators have conflicting aims.
- Some pre-processing algorithms solve this problem and permit smoothing and edge enhancement simultaneously.
- Another classification of local pre-processing methods is according to the transformation properties.
- **Linear** and **nonlinear** transformations can be distinguished.
- Linear operations calculate the resulting value in the output image pixel g(i,j) as a linear combination of brightnesses in a local neighborhood of the pixel f(i,j) in the input image.
- The contribution of the pixels in the neighborhood is weighted by coefficients h

$$f(i,j) = \sum_{(m,n)} \sum_{\in O} h(i-m,\ j-n)\ g(m,n) \qquad (4.24)$$

- The above equation is equivalent to discrete convolution with the kernel h, that is called a **convolution mask**.
- Rectangular neighborhoods O are often used with an odd number of pixels in rows and columns, enabling the specification of the central pixel of the neighborhood.
- Local pre-processing methods typically use very little a priori knowledge about the image contents. It is very difficult to infer this knowledge while an image is processed as the known neighborhood O of the processed pixel is small.
- The choice of the local transformation, size, and shape of the neighborhood O depends strongly on the size of objects in the processed image.
- If objects are rather large, an image can be enhanced by smoothing of small degradations.

Image smoothing:

- Image smoothing is the set of local pre-processing methods which have the aim of suppressing image noise - it uses redundancy in the image data.
- Calculation of the new value is based on averaging of brightness values in some neighborhood O.

- Smoothing poses the problem of blurring sharp edges in the image, and so we shall concentrate on smoothing methods which are **edge preserving**. They are based on the general idea that the average is computed only from those points in the neighborhood which have similar properties to the processed point.
- Local image smoothing can effectively eliminate impulsive noise or degradations appearing as thin stripes, but does not work if degradations are large blobs or thick stripes.

Averaging:

- Assume that the noise value at each pixel is an independent random variable with zero mean and standard deviation **σ**
- We can obtain such an image by capturing the same static scene several times.
- The result of smoothing is an average of the same n points in these images $g_1,...,g_{\{n\}}$ with noise values $\nu_{\{1\}},..., \nu_{\{n\}}$

$$\frac{g_1 + \ldots + g_n}{n} + \frac{\nu_1 + \ldots + \nu_n}{n} \qquad (4.25)$$

- The second term here describes the effect of the noise ... a random value with zero mean.
- Thus if n images of the same scene are available, the smoothing can be accomplished without blurring the image by

$$f(i,j) = \frac{1}{n}\sum_{k=1}^{n} g_k(i,j) \qquad (4.26)$$

- In many cases only one image with noise is available, and averaging is then realized in a local neighborhood.
- Results are acceptable if the noise is smaller in size than the smallest objects of interest in the image, but blurring of edges is a serious disadvantage.
- Averaging is a special case of discrete convolution. For a 3 x 3 neighborhood the convolution mask h is

$$h = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad (4.27)$$

- Larger convolution masks for averaging are created analogously.
- The significance of the central pixel may be increased to better reflect properties of Gaussian noise.

$$h = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad h = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad (4.28)$$

Averaging with limited data validity:

- Methods that average with limited data validity try to avoid blurring by averaging only those pixels which satisfy some criterion, the aim being to prevent involving pixels that are part of a separate feature.
- A very simple criterion is to use only pixels in the original image with brightness in a predefined interval of invalid data [min,max].
- Consider a point (m,n) in the image. If the intensity at (m,n) has a valid intensity, then nothing is done.
- However, if a point (m,n) has an invalid gray-level, then the convolution mask is calculated in the neighborhood O from the nonlinear formula

$$h(i,j) = \begin{cases} 1 & \text{for } g(m+i, n+j) \notin [min, max] \\ 0 & \text{otherwise} \end{cases} \qquad (4.29)$$

Note that in the equation above, the interval min-max represents invalid data so that only valid data is used in the averaging. Note that h (i,j) must be normalized by the number of data values used in the mask. A second method performs the averaging only if the computed brightness change of a pixel is in some predefined interval.

- This method permits repair to large-area errors resulting from slowly changing brightness of the background without affecting the rest of the image.
- A third method uses edge strength (i.e., magnitude of a gradient) as a criterion.
- The magnitude of some gradient operator is first computed for the entire image, and only pixels in the input image with a gradient magnitude smaller than a predefined threshold are used in averaging.

Averaging according to inverse gradient:

- The convolution mask is calculated at each pixel according to the inverse gradient.
- Brightness change within a region is usually smaller than between neighboring regions.
- Let (i,j) be the central pixel of a convolution mask with odd size; the inverse gradient at the point (m,n) with respect to (i,j) is then

$$\delta(i,j,m,n) = \frac{1}{|g(m,n) - g(i,j)|} \qquad (4.30)$$

- If g(m,n) = g(i,j) then we define δ(i,j,m,n) = 2;
- the inverse gradient is then in the interval (0,2], and is smaller on the edge than in the interior of a homogeneous region.
- Weight coefficients in the convolution mask h are normalized by the inverse gradient, and the whole term is multiplied by 0.5 to keep brightness values in the original range.
- The constant 0.5 has the effect of assigning half the weight to the central pixel (i,j), and the other half to its neighborhood.

$$h(i,j,m,n) = 0.5 \frac{\delta(i,j,m,n)}{\sum_{(m,n)\in\mathcal{O}} \delta(i,j,m,n)} \qquad (4.31)$$

- The convolution mask coefficient corresponding to the central pixel is defined as h(i,j) = 0.5.
- The above method assumes sharp edges.
- When the convolution mask is close to an edge, pixels from the region have larger coefficients than pixels near the edge, and it is not blurred. Isolated noise points within homogeneous regions have small values of the inverse gradient; points from the neighborhood take part in averaging and the noise is removed.

Averaging using a rotating mask:

- avoids edge blurring by searching for the homogeneous part of the current pixel neighborhood
- the resulting image is in fact sharpened
- brightness average is calculated only within the homogeneous region
- a brightness dispersion σ^2 is used as the region homogeneity measure.
- let n be the number of pixels in a region R and g(i,j) be the input image. Dispersion σ^2 is calculated as

$$\sigma^2 = \frac{1}{n}\left(\sum_{(i,j)\in R}\left(g(i,j) - \frac{1}{n}\sum_{(i,j)\in R} g(i,j)\right)^2\right) \qquad (4.32)$$

- The computational complexity (number of multiplications) of the dispersion calculation can be reduced if expressed as follows

$$
\begin{aligned}
\sigma^2 &= \frac{1}{n}\sum_{(i,j)\in R}\left((g(i,j))^2 - 2\,g(i,j)\frac{\sum_{(i,j)\in R} g(i,j)}{n} + \left(\frac{\sum_{(i,j)\in R} g(i,j)}{n}\right)^2\right) \\
&= \frac{1}{n}\left(\sum_{(i,j)\in R}(g(i,j))^2 - 2\,\frac{\left(\sum_{(i,j)\in R} g(i,j)\right)^2}{n} + n\left(\frac{\sum_{(i,j)\in R} g(i,j)}{n}\right)^2\right) \\
&= \frac{1}{n}\left(\sum_{(i,j)\in R}(g(i,j))^2 - \frac{\left(\sum_{(i,j)\in R} g(i,j)\right)^2}{n}\right) \qquad (4.33)
\end{aligned}
$$

- Rotated masks

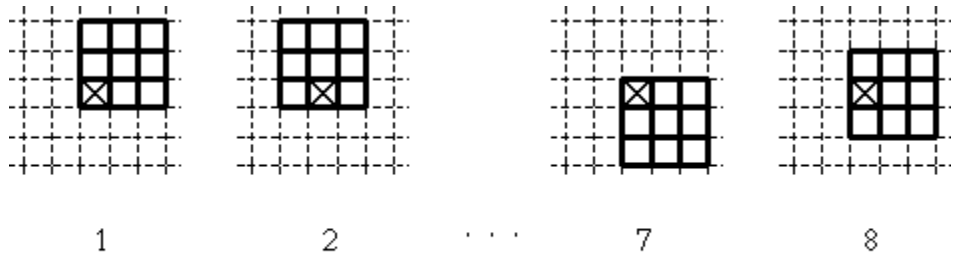


Figure 4.11 8 possible rotated 3×3 masks.

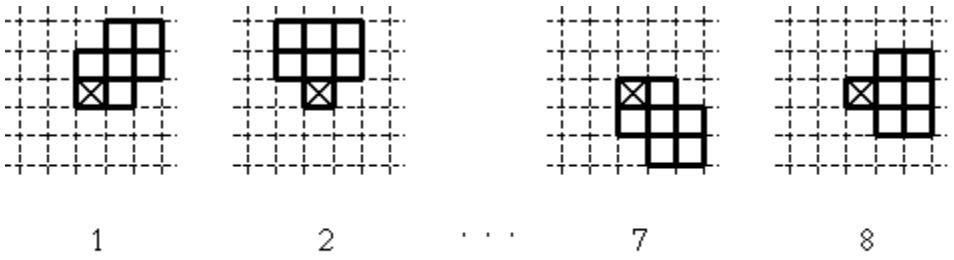


Figure 4.12 Alternative shape of 8 possible rotated masks.

Algorithm 4.1: Rotated mask smoothing

1. Consider each image pixel (i, j).
2. Calculate dispersion in the mask for all possible mask rotations about pixel (i, j) according to equation (4.32).
3. Choose the mask with minimum dispersion.
4. Assign to the pixel $g(i, j)$ in the output image the average brightness in the chosen mask.

Median smoothing:

- In a set of ordered values, the median is the central value.
- Median filtering reduces blurring of edges.
- The idea is to replace the current point in the image by the median of the brightness in its neighborhood.
- Not affected by individual noise spikes
- eliminates impulsive noise quite well
- Does not blur edges much and can be applied iteratively.
- The main disadvantage of median filtering in a rectangular neighborhood is its damaging of thin lines and sharp corners in the image -- this can be avoided if another shape of neighborhood is used.

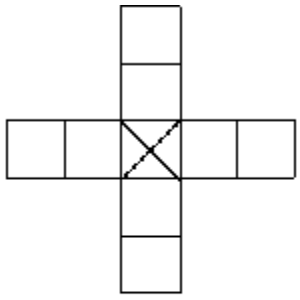


Figure 4.14 *Horizontal/vertical line-preserving neighbourhood for median filtering.*

Edge Detectors:

- Locate sharp changes in the intensity function
- Edges are pixels where brightness changes abruptly.
- Calculus describes changes of continuous functions using derivatives; an image function depends on two variables - partial derivatives.

- A change of the image function can be described by a gradient that points in the direction of the largest growth of the image function.
- An edge is a property attached to an individual pixel and is calculated from the image function behavior in a neighborhood of the pixel.
- It is a **vector variable**
    - **magnitude** of the gradient
    - **direction**
- The gradient direction gives the direction of maximal growth of the function, e.g., from black (f (i,j)=0) to white (f(i,j)=255).
- This is illustrated below; closed lines are lines of the same brightness.
- The orientation 0° points east.

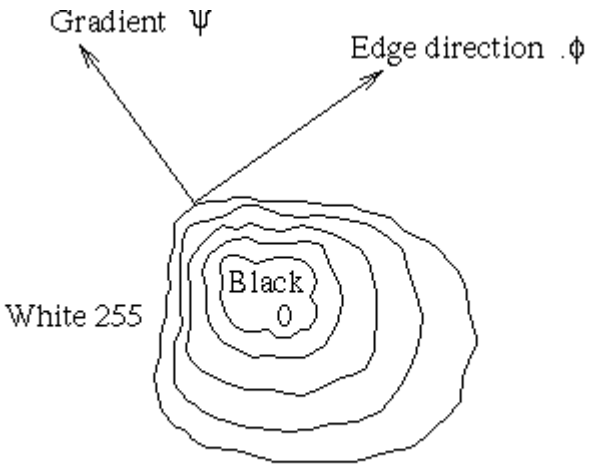


Figure 4.16 Gradient direction and edge direction.

- Edges are often used in image analysis for finding region boundaries.
- Boundary and its parts (edges) are perpendicular to the direction of the gradient.

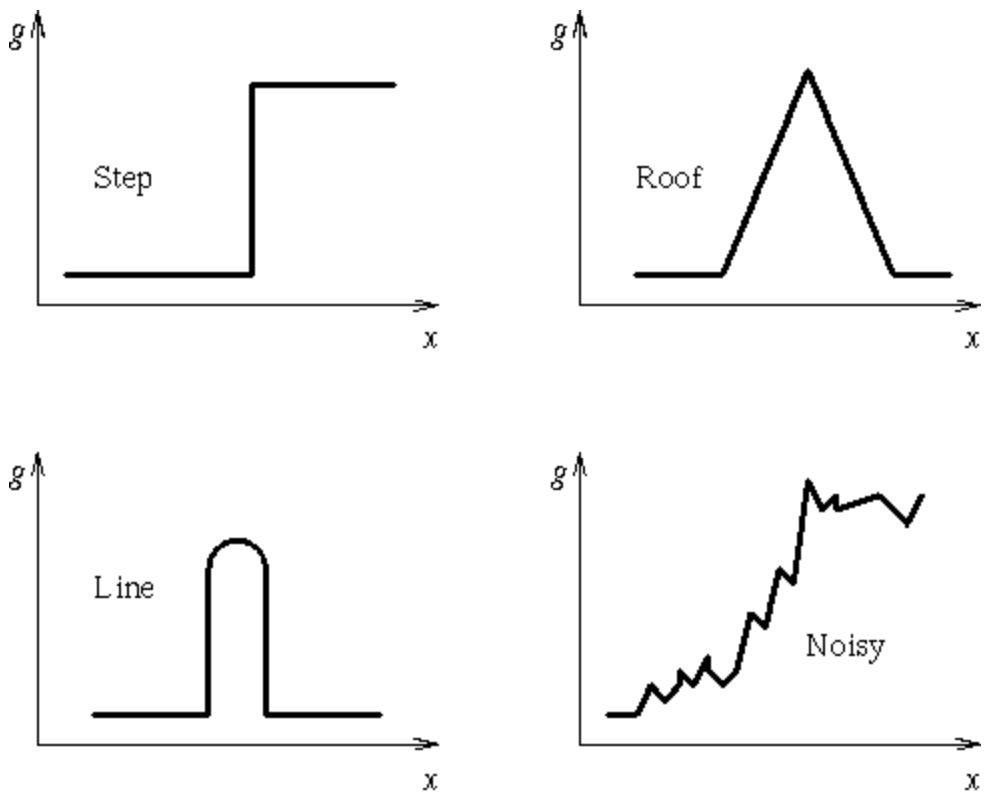


Figure 4.17 Typical edge profiles.

- The gradient magnitude and gradient direction are continuous image functions where arg(x,y) is the angle (in radians) from the x-axis to the point (x,y).

$$|grad\ g(x,y)| = \sqrt{\left(\frac{\partial g}{\partial x}\right)^2 + \left(\frac{\partial g}{\partial y}\right)^2} \qquad (4.35)$$

$$\psi = arg(\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}) \qquad (4.36)$$

- Sometimes we are interested only in edge magnitudes without regard to their orientations.
- The **Laplacian** may be used.
- The Laplacian has the same properties in all directions and is therefore invariant to rotation in the image.

$$\nabla^2 (x,y) = \frac{\partial^2 g(x,y)}{\partial x^2} + \frac{\partial^2 g(x,y)}{\partial y^2} \qquad (4.37)$$

- Image sharpening makes edges steeper -- the sharpened image is intended to be observed by a human.
- C is a positive coefficient which gives the strength of sharpening and S(i,j) is a measure of the image function sheerness that is calculated using a gradient operator.
- The Laplacian is very often used to estimate S(i,j).

$$f(i,j) = g(i,j) - C \;\; S(i,j) \qquad (4.38)$$

Laplace operator:

- The Laplace operator (Eq. 4.37) is a very popular operator approximating the second derivative which gives the gradient magnitude only.
- The Laplacian is approximated in digital images by a convolution sum.
- A 3 x 3 mask for 4-neighborhoods and 8-neighborhood

$$h = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad h = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad (4.43)$$

- A Laplacian operator with stressed significance of the central pixel or its neighborhood is sometimes used. In this approximation it loses invariance to rotation

$$h = \begin{bmatrix} 2 & -1 & 2 \\ -1 & -4 & -1 \\ 2 & -1 & 2 \end{bmatrix} \qquad h = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix} \qquad (4.44)$$

- The Laplacian operator has a disadvantage -- it responds doubly to some edges in the image.
- Image sharpening / edge detection can be interpreted in the frequency domain as well.
- The result of the Fourier transform is a combination of harmonic functions.
- The derivative of the harmonic function sin (nx) is n cos (nx); thus the higher the frequency, the higher the magnitude of its derivative.
- This is another explanation of why gradient operators enhance edges.
- **Unsharp masking** is often used in printing industry applications - another image sharpening approach.
- A signal proportional to an unsharp image (e.g., blurred by some smoothing operator) is subtracted from the original image, again a parameter C may be used to control the weight of the subtraction.
- A digital image is discrete in nature ... derivatives must be approximated by **differences**.

- The first differences of the image g in the vertical direction (for fixed i) and in the horizontal direction (for fixed j)

$$\Delta_i g(i,j) = g(i,j) - g(i-n,j)$$
$$\Delta_j g(i,j) = g(i,j) - g(i,j-n) \qquad (4.39)$$

- n is a small integer, usually 1.
- The value n should be chosen small enough to provide a good approximation to the derivative, but large enough to neglect unimportant changes in the image function.
- Symmetric expressions for the difference are not usually used because they neglect the impact of the pixel (i,j) itself.

$$\Delta_i g(i,j) = g(i+n,j) - g(i-n,j)$$
$$\Delta_j g(i,j) = g(i,j+n) - g(i,j-n) \qquad (4.40)$$

- Gradient operators can be divided into **three** categories

  I. Operators approximating derivatives of the image function using differences.
  - Rotationally invariant (e.g., Laplacian) need one convolution mask only.
  - Approximating first derivatives use several masks ... the orientation is estimated on the basis of the best matching of several simple patterns.

  II. Operators based on the zero crossings of the image function second derivative (e.g., Marr-Hildreth or Canny edge detector).

  III. Operators which attempt to match an image function to a parametric model of edges.

This category will not be covered here; parametric models describe edges more precisely than simple edge magnitude and direction and are much more computationally intensive. Individual gradient operators that examine small local neighborhoods are in fact convolutions and can be expressed by convolution masks. Operators which are able to detect edge direction as well are represented by a collection of masks, each corresponding to a certain direction.

Roberts's operator:

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad (4.41)$$

- so the magnitude of the edge is computed as

$$|g(i,j) - g(i+1,j+1)| + |g(i,j+1) - g(i+1,j)| \qquad (4.42)$$

- The primary disadvantage of the Roberts operator is its high sensitivity to noise, because very few pixels are used to approximate the gradient.

Prewitt operator:

- The Prewitt operator, similarly to the Sobel, Kirsch, Robinson (as discussed later) and some other operators, approximates the first derivative.
- Operators approximating first derivative of an image function are sometimes called compass operators because of the ability to determine gradient direction.
- The gradient is estimated in eight (for a 3 x 3 convolution mask) possible directions, and the convolution result of greatest magnitude indicates the gradient direction. Larger masks are possible.

$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} h_2 = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix} h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad (4.45)$$

- The direction of the gradient is given by the mask giving maximal response. This is valid for all following operators approximating the first derivative.

Sobel operator:

$$h_1 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} h_2 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad (4.46)$$

- Used as a simple detector of horizontality and verticality of edges in which case only masks h_1 and h_3 are used.
- If the h_1 response is y and the h_3 response x, we might then derive edge strength (magnitude) as

$$\sqrt{x^2 + y^2} \qquad or \qquad |x| + |y| \qquad (4.47)$$

and direction as arctan (y / x).

Robinson operator:

$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix} \quad h_2 = \begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix} \quad h_3 = \begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix} \quad (4.48)$$

Kirsch operator:

$$h_1 = \begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix} \quad h_2 = \begin{bmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{bmatrix} \quad h_3 = \begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix} \quad (4.49)$$

IMAGE RESTORATION:

- **Image restoration** - suppressing image degradation using knowledge about its nature

Image restoration as inverse convolution of the whole image:

- Most image restoration methods are based on convolution applied globally to the whole image.
- Degradation causes:
  - defects of optical lenses,
  - nonlinearity of the electro-optical sensor,
  - graininess of the film material,
  - relative motion between an object and camera
  - wrong focus,
  - atmospheric turbulence in remote sensing or astronomy,
- The objective of image restoration is to reconstruct the original image from its degraded version.
- Image restoration techniques - two groups:
- **Deterministic** methods - applicable to images with little noise and a known degradation function.
  - The original image is obtained from the degraded one by a transformation inverse to the degradation.
- **Stochastic** techniques - the best restoration is sought according to some stochastic criterion, e.g., a least squares method.
  - In some cases the degradation transformation must be estimated first.
- It is advantageous to know the degradation function explicitly.
- The better this knowledge is, the better are the results of the restoration.

- There are three typical degradations with a simple function:
  - Relative constant speed movement of the object with respect to the camera,
  - wrong lens focus,
  - Atmospheric turbulence.
- In most practical cases, there is insufficient knowledge about the degradation, and it must be estimated and modeled.
- Methods:
  - **A priori knowledge** about degradation - either known in advance or obtained before restoration.
  - If it is clear in advance that the image was degraded by relative motion of an object with respect to the sensor then the modeling only determines the speed and direction of the motion.
- An example of parameters obtained before restoration is an attempt to estimate parameters of a capturing device such as a TV camera or digitizer.
- **A posteriori knowledge** is obtained by analyzing the degraded image.
- A typical example is to find some interest points in the image (e.g. corners, straight lines) and guess how they looked before degradation.
- Another possibility is to use spectral characteristics of the regions in the image that are relatively homogeneous.
- A degraded image g can arise from the original image f by a process which can be expressed as

$$g(i,j) = s\left(\int\int_{(a,b)\in\mathcal{O}} f(a,b)\, h(a,b,i,j)\, da\, db\right) + \nu(i,j) \qquad (4.77)$$

where s is some nonlinear function and nu describes the noise.

- The degradation is very often simplified by
  - neglecting the nonlinearity
  - Assuming that the function h is invariant with respect to position in the image.
- Degradation can be then expressed as convolution

$$g(i,j) = (f * h)(i,j) + \nu(i,j) \qquad (4.78)$$

- If the degradation is given by equation (4.78) and the noise is not significant then image restoration equates to **inverse convolution** (also called **deconvolution**).

- If noise is not negligible then the inverse convolution is solved as an overdetermined system of linear equations.

Degradations that is easy to restore:

- Some degradations can be easily expressed mathematically (convolution) and also restored simply in images.
- The Fourier transform H of the convolution function is used.

Relative motion of the camera and object:

- Assume an image is acquired with a camera with a mechanical shutter.
- Relative motion of the camera and the photographed object during the shutter open time T causes smoothing of the object in the image.
- Suppose V is the constant speed in the direction of the x axis; the Fourier transform H(u,v) of the degradation caused in time T is given by

$$H(u,v) = \frac{\sin(\pi V T u)}{\pi V u} \qquad (4.79)$$

Wrong lens focus:

- Image smoothing caused by imperfect focus of a thin lens can be described by the following function

$$H(u,v) = \frac{J_1(a\,r)}{a\,r} \qquad (4.80)$$

where J_1 is the Bessel function of the first order, r^2 = u^2 + v^2, and a is the displacement.

Atmospheric turbulence:

- needs to be restored in remote sensing and astronomy
- Caused by temperature non-homogeneity in the atmosphere that deviates passing light rays.
- The mathematical model

$$H(u,v) = e^{-c(u^2+v^2)^{\frac{5}{6}}} \qquad (4.81)$$

Where c is a constant that depends on the type of turbulence which is usually found experimentally. The power 5/6 is sometimes replaced by 1.

Inverse filtration:

- based on the assumption that degradation was caused by a linear function h(i,j)
- the additive noise nu is another source of degradation.
- It is further assumed that nu is independent of the signal.

- Applying the Fourier transform to equation (4.78)

$$G(u,v) = F(u,v)\ H(u,v) + N(u,v) \qquad (4.82)$$

- The degradation can be eliminated if the restoration filter has a transfer function that is inverse to the degradation h ... or in Fourier transform H^-1(u,v).
- The undegraded image F is derived from its degraded version G.

$$F(u,v) = G(u,v)\ H^{-1}(u,v) - N(u,v)\ H^{-1}(u,v) \qquad (4.83)$$

- This equation shows that inverse filtration works well for images that are not corrupted by noise.
- If noise is present and additive error occurs, its influence is significant for frequencies where H(u,v) has small magnitude. These usually correspond to high frequencies u,v and thus fine details are blurred in the image.
- The changing level of noise may cause problems as well because small magnitudes of H(u,v) can cause large changes in the result.
- Inverse filter values should not be of zero value so as to avoid zero divides in equation (4.83).

Wiener filtration:

- It is no surprise that inverse filtration gives poor results in pixels suffering from noise since the noise is not taken into account.
- Wiener filtration explores a priori knowledge about the noise.
- Restoration by the Wiener filter gives an estimate of the original uncorrupted image f with minimal mean square error e^2

$$e^2 = \mathcal{E}\left\{(f(i,j) - \hat{f}(i,j))^2\right\} \qquad (4.84)$$

- No constraints applied to equation (4.84)
  - optimal estimate ^f is the conditional mean value of the ideal image f under the condition g
  - complicated from the computational point of view
  - The conditional probability density between the optimal image f and the corrupted image g is not usually known.
  - The optimal estimate is in general a nonlinear function of the image g.
- Minimization of equation (4.84) is easy if estimate ^f is a linear combination of the values in the image g -close-optimal solution
- H_W ... Fourier transform of the Wiener filter - considers noise - see Eq. (4.86) below

- G ... Fourier transform of the degraded image

$$\hat{F}(u,v) = H_W(u,v)\, G(u,v) \qquad (4.85)$$

$$H_W(u,v) = \frac{H^{\#}(u,v)}{|H(u,v)|^2 + \frac{S_{\nu\nu}(u,v)}{S_{gg}(u,v)}} \qquad (4.86)$$

where H is the transform function of the degradation
# denotes complex conjugate
S_{nu nu} is the spectral density of the noise
S_{gg} is the spectral density of the degraded image

- If Wiener filtration is used, the nature of degradation H and statistical parameters of the noise need to be known.
- Wiener filtration theory solves the problem of a posteriori linear mean square estimate -- all statistics (e.g., power spectrum) should be available in advance. Note that the inverse filter discussed earlier is a special case of the Wiener filter where noise is absent i.e. S_{nu nu} = 0.

QUESTION BANK

5 MARKS:

1. Explain the Topological data structures in detail **(April 2012)**.
2. Explain the Hierarchical data structures in detail.
3. Explain the local pre-processing in detail
4. Explain the types of operator in detail **(April 2012)**.

10 MARKS:

1. Explain the Levels of image data representation in detail **(April 2012)**.
2. Explain the Geometric transformations in detail **(April 2012)**.
3. Explain the Image restoration in detail.

UNIT-III

SEGMENTATION:

- One of the most important steps leading to the analysis of processed image data. its main goal is to divide an image into parts that have a strong correlation with objects or areas of the real world contained in the image
- Complete segmentation - set of disjoint regions uniquely corresponding with objects in the input image. Cooperation with higher processing levels which use specific knowledge of the problem domain is necessary.
- Partial segmentation - regions do not correspond directly with image objects. Image is divided into separate regions that are homogeneous with respect to a chosen property such as brightness, color, reflectivity, texture, etc.

- In a complex scene, a set of possibly overlapping homogeneous regions may result. The partially segmented image must then be subjected to further processing, and the final image segmentation may be found with the help of higher level information.
- Simple segmentation problems:
  - Contrasted objects on a uniform background
  - Simple assembly tasks, blood cells, printed characters, etc.
- Totally correct and complete segmentation of complex scenes usually cannot be achieved in this processing phase.
- A reasonable aim is to use partial segmentation as an input to higher level processing.
- **Segmentation problems:**
  - image data ambiguity
  - information noise
- **Segmentation methods:**
  - global approaches, e.g. using histogram of image features
  - edge-based segmentations
  - region-based segmentations
- Characteristics used in edge detection or region growing
  - brightness
  - texture
  - velocity field
- Edge-based and region-bases segmentation approaches solve a dual problem ... border x region.
- Because of the different natures of the various edge- and region-based algorithms, they may be expected to give somewhat different results and consequently different information.
- The segmentation results of these two approaches can therefore be combined in a single description structure, e.g., a relational graph.

THRESHOLDING:

Gray level thresholding is the simplest segmentation process. Many objects or image regions are characterized by constant reflectivity or light absorption of their surface.

Thresholding is computationally inexpensive and fast. Thresholding can easily be done in real time using specialized hardware. Complete segmentation can result from thresholding in simple scenes.

$$R = \bigcup_{i=1}^{S} R_i \qquad R_i \cap R_j = \emptyset \qquad i \neq j \qquad (5.1)$$

- Thresholding

$$g(i,j) = 1 \quad \text{for } f(i,j) \geq T$$
$$= 0 \quad \text{for } f(i,j) < T \qquad (5.2)$$

1. Thresholding algorithm:

Basic thershlding

- Search all the pixels f(i,j) of the image f. An image element g(i,j) of the segmented image is an object pixel if f(i,j) >= T, and is a background pixel otherwise.

- Correct threshold selection is crucial for successful threshold segmentation.
- Threshold selection can be interactive or can be the result of some threshold detection method.
- Single global threshold ... successful only under very unusual circumstances
  - Gray level variations are likely - due to non-uniform lighting, non-uniform input device parameters or a number of other factors.

$$T = T(f) \qquad (5.3)$$

- **Variable thresholding** (also **adaptive thresholding**), in which the threshold value varies over the image as a function of local image characteristics, can produce the solution in these cases.

  - image f is divided into subimages f_c
  - a threshold is determined independently in each subimage
  - If a threshold cannot be determined in some subimage, it can be interpolated from thresholds determined in neighboring subimages.
  - Each subimage is then processed with respect to its local threshold.

$$T = T(f, f_c) \qquad (5.4)$$

2. Thresholding modifications:

- **Band-thresholding:**

  - segment an image into regions of pixels with gray levels from a set D and into background otherwise

$$\begin{aligned} g(i,j) &= 1 \quad \text{for } f(i,j) \in D \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{5.5}$$

- Can also serve as border detection.

- **Multi thresholding:**
  - Resulting image is no longer binary.

$$\begin{aligned} g(i,j) &= 1 \quad \text{for } f(i,j) \in D_1 \\ &= 2 \quad \text{for } f(i,j) \in D_2 \\ &= 3 \quad \text{for } f(i,j) \in D_3 \\ &= 4 \quad \text{for } f(i,j) \in D_4 \\ &\cdots \\ &= n \quad \text{for } f(i,j) \in D_n \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{5.6}$$

- **Semi thresholding:**

  Aims to mask out the image background leaving gray level information present in the objects

$$\begin{aligned} g(i,j) &= f(i,j) \quad \text{for } f(i,j) \geq T \\ &= 0 \quad \text{for } f(i,j) < T \end{aligned} \tag{5.7}$$

- Thresholding can also be applied to
  - gradient
  - local texture
  - any other image decomposition criterion

3. Threshold detection methods:

- If some property of an image after segmentation is known a priori, the task of threshold selection is simplified, since the threshold is chosen to ensure this property is satisfied.
- A printed text sheet may be an example if we know that characters of the text cover 1/p of the sheet area.

a. P-tile-thresholding:

- Choose a threshold T (based on the image histogram) such that 1/p of the image area has gray values less than T and the rest has gray values larger than T.
- In text segmentation, prior information about the ratio between the sheet area and character area can be used.
- If such a priori information is not available - another property, for example the average width of lines in drawings, etc. can be used - the threshold can be determined to provide the required line width in the segmented image.

b. More complex methods of threshold detection:

- Based on histogram shape analysis.
- **Bimodal histogram** - if objects have approximately the same gray level that differs from the gray level of the background.

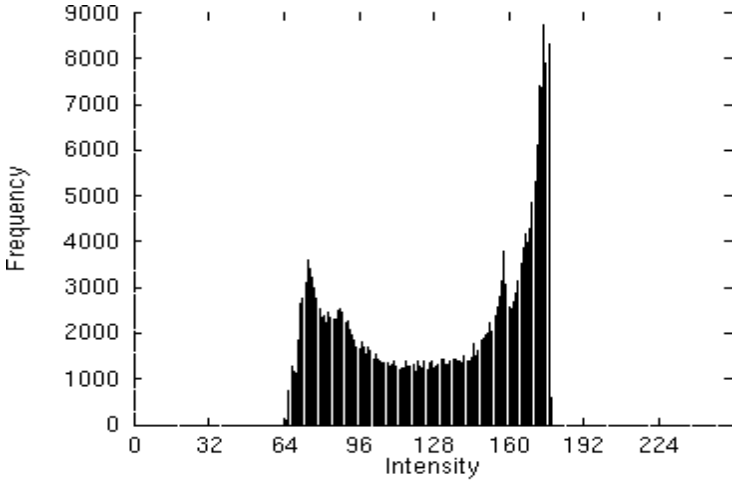


Figure 5.3 *A bimodal histogram.*

- Threshold-based segmentation ... minimum segmentation error requirements

- It makes intuitive sense to determine the threshold as the gray level that has a minimum histogram value between the two mentioned maxima
- Multimodal histogram - more thresholds may be determined at minima between any two maxima.

c. Bimodality of histograms:

- To decide if a histogram is bimodal or multimodal may not be so simple in reality.
- It is often impossible to interpret the significance of local histogram maxima.

- Bimodal histogram threshold detection algorithms
  - **Mode method** - find the highest local maxima first and detect the threshold as a minimum between them
    - to avoid detection of two local maxima belonging to the same global maximum, a minimum distance in gray levels between these maxima is usually required
    - Techniques to smooth histograms are applied.
- Histogram bimodality itself does not guarantee correct threshold segmentation.

4. Optimal thresholding:

Based on approximation of the histogram of an image using a weighted sum of two or more probability densities with normal distribution. The threshold is set as the closest gray level corresponding to the minimum probability between the maxima of two or more normal distributions, which results in minimum error segmentation

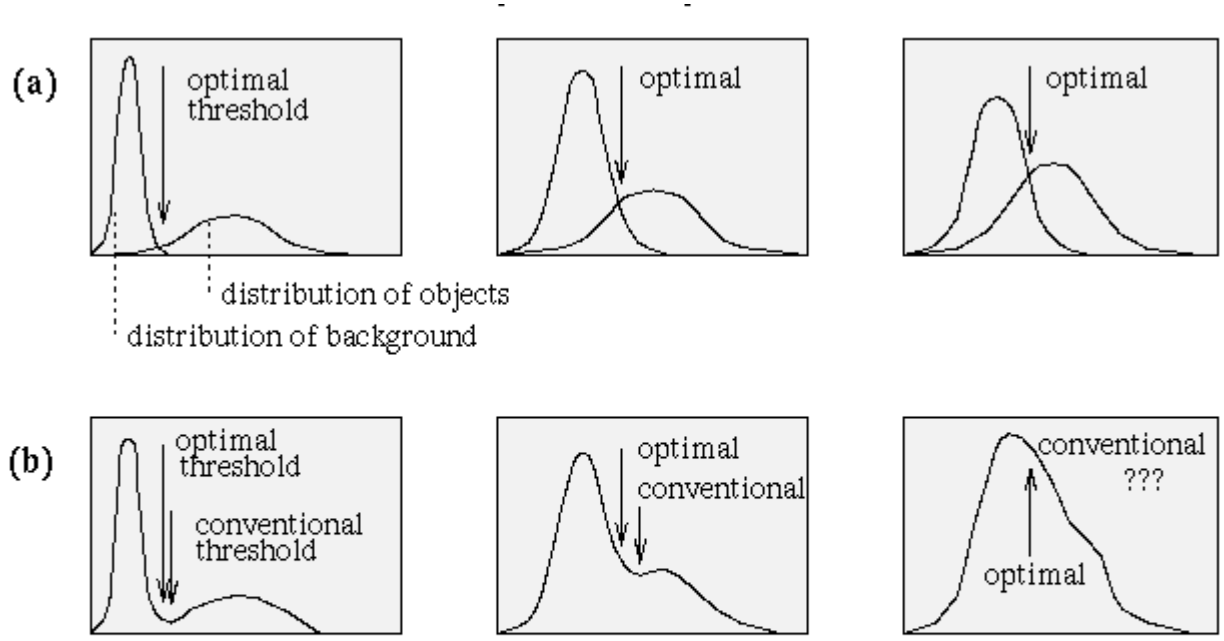


Figure 5.4 *Grey level histograms approximated by two normal distributions; the threshold is set to give minimum probability of segmentation error: (a) Probability distributions of background and objects, (b) corresponding histograms and optimal threshold.*

- Problems - estimating normal distribution parameters together with the uncertainty that the distribution may be considered normal.

A. Motivation algorithm:

Algorithm 5.2: Iterative (optimal) threshold selection

1. Assuming no knowledge about the exact location of objects, consider as a first approximation that the four corners of the image contain background pixels only and the remainder contains object pixels.

2. At step t, compute μ_B^t and μ_O^t as the mean background and object grey level respectively, where segmentation into background and objects at step t is defined by the threshold value T^t determined in the previous step (equation 5.9);

$$\mu_B^t = \frac{\sum_{(i,j)\in background} f(i,j)}{\#background\_pixels} \qquad \mu_O^t = \frac{\sum_{(i,j)\in objects} f(i,j)}{\#object\_pixels} \tag{5.8}$$

3. Set

$$T^{(t+1)} = \frac{\mu_B^t + \mu_O^t}{2} \tag{5.9}$$

$T^{(t+1)}$ now provides an updated background/object distinction.

4. If $T^{(t+1)} = T^{(t)}$, halt; otherwise return to (2).

- The method performs well under a large variety of image contrast conditions

B. Example - Brain MR Image segmentation:

- A combination of optimal and adaptive thresholding
  - determines optimal gray level segmentation parameters in local sub regions for which local histograms are constructed
  - gray-level distributions corresponding to n individual (possibly non-contiguous) regions are fitted to each local histogram that is modeled as a sum of n Gaussian distributions so that the difference between the modeled and the actual histograms is minimized

$$h_{model}(g) = \sum_{i=1}^{n} a_i e^{(-\frac{(g-\mu_i)^2}{2\sigma_i^2})} \tag{5.10}$$

- Variable g represents gray level values from the set G of images gray levels, a_i, &sigma$_i$ and μ_i denote parameters of the Gaussian distribution for the region i.

- The optimal parameters of the Gaussian distributions are determined by minimizing the fit function F

$$F = \sum_{g \in G} (h_{model}(g) - h_{region}(g))^2 \qquad (5.11)$$

- Applied to segmentation of MR brain images, three segmentation classes - WM, GM, CSF

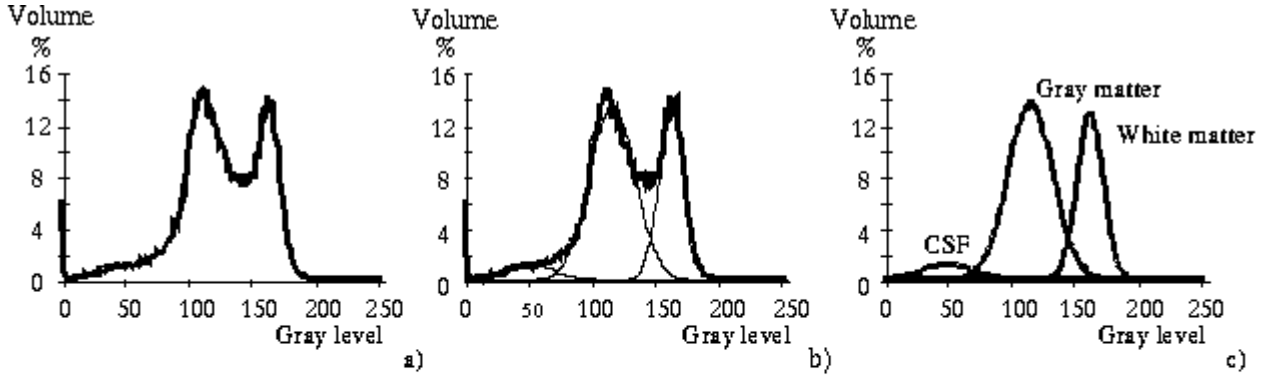


Figure 5.5 *Segmentation of 3-D T1-weighted MR brain image data using optimal thresholding: (a) Local gray level histogram, (b) fitted Gaussian distributions; global 3-D image fit, (c) Gaussian distributions corresponding to WM, GM, and CSF. Courtesy R.J. Frank, T.J. Grabowski, Human Neuroanatomy and Neuroimaging Laboratory, Department of Neurology, The University of Iowa.*

5. Multi-spectral thresholding:

Multispectral or color images. One segmentation approach determines thresholds independently in each spectral band and combines them into a single segmented image.

Algorithm 5.3: Recursive multispectral thresholding

1. Initialize the whole image as a single region.

2. Compute a smoothed histogram (see Section ??) for each spectral band. Find the most significant peak in each histogram and determine two thresholds as local minima on either side of this maximum. Segment each region in each spectral band into subregions according to these thresholds. Each segmentation in each spectral band is projected into a multispectral segmentation – see Figure 5.7. Regions for the next processing steps are those in the multispectral image.

3. Repeat step (2) for each region of the image until each region's histogram contains only one significant peak.

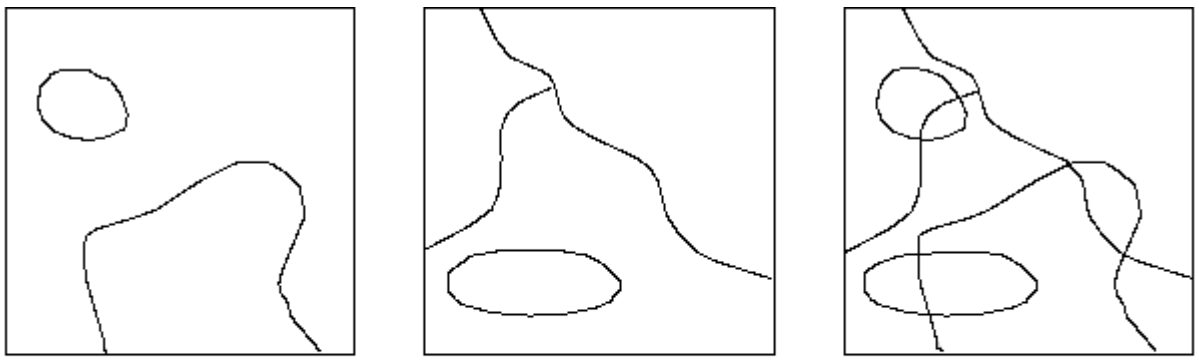


Figure 5.7 *Recursive multispectral thresholding: (a) Band 1 thresholding, (b) band 2 thresholding, (c) multispectral segmentation.*

6. Hierarchical Thresholding:

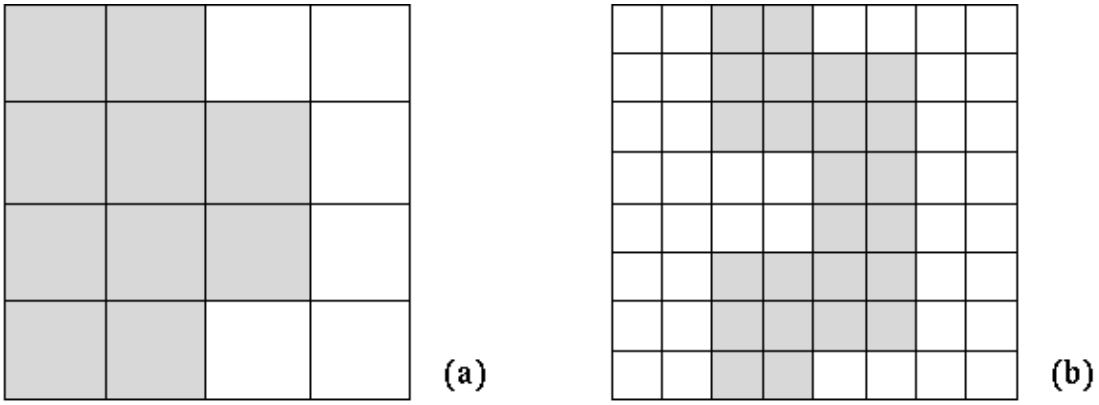


Figure 5.8 *Hierarchical thresholding: (a) Pyramid level n, segmentation to objects and background, (b) pyramid level n−1, showing where the thresholding must be repeated for better precision.*

$$T = \frac{c + \frac{1}{g}\sum_i n_i}{2} \qquad (5.12)$$

EDGE-BASED SEGMENTATION:

Edge-based segmentation represents a large group of methods based on information about edges in the image. Edge-based segmentations rely on edges found in an image by edge detecting operators -- these edges mark image locations of discontinuities in gray level, color, texture, etc.

Image resulting from edge detection cannot be used as a segmentation result. Supplementary processing steps must follow to combine edges into edge chains that correspond better with borders in the image.

The final aim is to reach at least a partial segmentation -- that is, to group local edges into an image where only edge chains with a correspondence to existing objects or image parts are present. The more prior information that is available to the segmentation process, the better the segmentation results that can be obtained. The most common problems of edge-based segmentation are

- an edge presence in locations where there is no border, and
- No edge presence where a real border exists.

A. EDGE IMAGE THRESHOLDING:

- Almost no zero-value pixels are present in an edge image, but small edge values correspond to non-significant gray level changes resulting from quantization noise, small lighting irregularities, etc.

- Selection of an appropriate global threshold is often difficult and sometimes impossible; p-tile thresholding can be applied to define a threshold
- Alternatively, non-maximal suppression and hysteresis thresholding can be used as was introduced in the Canny edge detector.

B. EDGE RELAXATION:

- Borders resulting from the previous method are strongly affected by image noise, often with important parts missing. Considering edge properties in the context of their mutual neighbors can increase the quality of the resulting image.
- All the image properties, including those of further edge existence, are iteratively evaluated with more precision until the edge context is totally clear - based on the strength of edges in a specified local neighborhood, the confidence of each edge is either increased or decreased.
- A weak edge positioned between two strong edges provides an example of context; it is highly probable that this inter-positioned weak edge should be a part of a resulting boundary. If, on the other hand, an edge (even a strong one) is positioned by itself with no supporting context, it is probably not a part of any border.

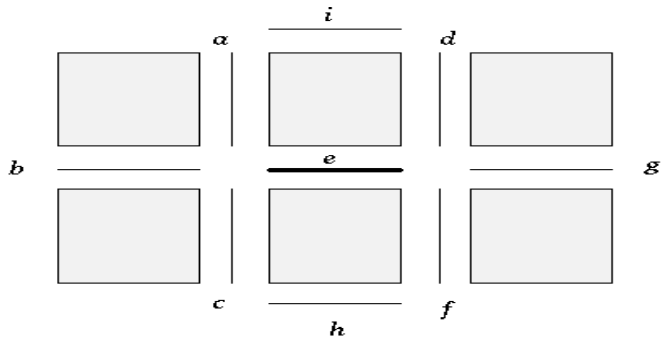


Figure 5.10 *Crack edges surrounding central edge e.*

- Edge context is considered at both ends of an edge, giving the minimal edge neighborhood.
- The central edge e has a vertex at each of its ends and three possible border continuations can be found from both of these vertices.
- Vertex type -- number of edges emanating from the vertex, not counting the edge e.
- The type of edge e can then be represented using a number pair i-j describing edge patterns at each vertex, where i and j are the vertex types of the edge e.

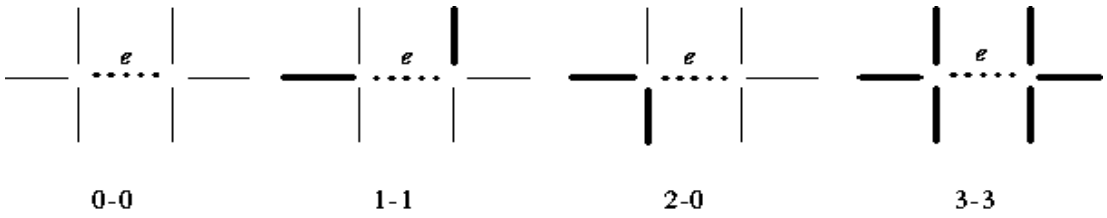


Figure 5.11 *Edge patterns and corresponding edge types.*

- **0-0** isolated edge – negative influence on the edge confidence
- **0-2, 0-3** dead end – negative influence on edge confidence
- **0-1** uncertain – weak positive, or no influence on edge confidence
- **1-1** continuation – strong positive influence on edge confidence
- **1-2, 1-3** continuation to border intersection – medium positive influence on edge confidence
- **2-2, 2-3, 3-3** bridge between borders – not necessary for segmentation, no influence on edge confidence

- Edge relaxation is an iterative method, with edge confidences converging either to zero (edge termination) or one (the edge forms a border).
- The confidence of each edge e in the first iteration can be defined as a normalized magnitude of the crack edge.
  - with normalization based on either the global maximum of crack edges in the whole image, or on a local maximum in some large neighborhood of the edge

Algorithm 5.5: Edge relaxation

1. Evaluate a confidence $c^{(1)}(e)$ for all crack edges e in the image.
2. Find the edge type of each edge based on edge confidences in its neighborhood.
3. Update the confidence $c^{(k+1)}(e)$ of each edge e according to its type and its previous confidence $c^{(k)}(e)$.
4. Stop if all edge confidences have converged either to 0 or 1. Repeat steps (2) and (3) otherwise.

- The main steps of the above Algorithm are evaluation of vertex types followed by evaluation of edge types, and the manner in which the edge confidences are modified.
- A vertex is considered to be of type i if

$$type(i) = \max_k(type(k)), \qquad k = 0, 1, 2, 3 \qquad (5.13)$$

$$\begin{aligned} type(0) &= (m-a)(m-b)(m-c) \\ type(1) &= a(m-b)(m-c) \\ type(2) &= ab(m-c) \\ type(3) &= abc \end{aligned}$$

- where a,b,c are the normalized values of the other incident crack edges
- m=max (a,b,c,q) ... the introduction of the quantity q ensures that type(0) is non-zero for small values of a.
- For example, choosing q=0.1, a vertex (a,b,c)=(0.5, 0.05, 0.05) is a type 1 vertex, while a vertex (0.3, 0.2, 0.2) is a type 3 vertex.
- Similar results can be obtained by simply counting the number of edges emanating from the vertex above a threshold value.
- Edge type is found as a simple concatenation of vertex types, and edge confidences are modified as follows:

$$\text{confidence increase:} \qquad c^{(k+1)}(e) = \min(1, c^{(k)}(e) + \delta) \qquad (5.14)$$

$$\text{confidence decrease:} \qquad c^{(k+1)}(e) = \max(0, c^{(k)}(e) - \delta) \qquad (5.15)$$

- Edge relaxation, as described above, rapidly improves the initial edge labeling in the first few iterations.
- Unfortunately, it often slowly drifts giving worse results than expected after larger numbers of iterations.
- The reason for this strange behavior is in searching for the global maximum of the edge consistency criterion over all the image, which may not give locally optimal results.
- A solution is found in setting edge confidences to zero under a certain threshold, and to one over another threshold which increases the influence of original image data.
- Therefore, one additional step must be added to the edge confidence computation

$$\text{if} \quad c^{(k+1)}(e) > T_1 \quad \text{then assign} \quad c^{(k+1)}(e) = 1 \qquad (5.16)$$

$$\text{if} \quad c^{(k+1)}(e) < T_2 \quad \text{then assign} \quad c^{(k+1)}(e) = 0 \qquad (5.17)$$

- Where T_1 and T_2 are parameters controlling the edge relaxation convergence speed and resulting border accuracy.
- This method makes multiple labeling possible; the existence of two edges at different directions in one pixel may occur in corners, crosses, etc.
- The relaxation method can easily be implemented in parallel.

C. BORDER TRACING:

- If a region border is not known but regions have been defined in the image, borders can be uniquely detected.

- First, let us assume that the image with regions is either binary or that regions have been labeled.
- An **inner** region border is a subset of the region
- An **outer** border is not a subset of the region.
- The following algorithm covers inner boundary tracing in both 4-connectivity and 8-connectivity.

Algorithm 5.6: Inner boundary tracing

1. Search the image from top left until a pixel of a new region is found; this pixel P_0 then has the minimum column value of all pixels of that region having the minimum row value. Pixel P_0 is a starting pixel of the region border. Define a variable *dir* which stores the direction of the previous move along the border from the previous border element to the current border element. Assign

   (a) $dir = \mathbf{0}$ if the border is detected in 4-connectivity (Figure 5.13a)

   (b) $dir = 7$ if the border is detected in 8-connectivity (Figure 5.13b)

2. Search the 3×3 neighborhood of the current pixel in an anti-clockwise direction, beginning the neighborhood search in the pixel positioned in

   the direction

   (a) $(dir + 3)$ *mod* 4 (Figure 5.13c)

   (b) $(dir + 7)$ *mod* 8 if *dir* is *even* (Figure 5.13d)
   $(dir + 6)$ *mod* 8 if *dir* is *odd* (Figure 5.13e)

   The first pixel found with the same value as the current pixel is a new boundary element P_n. Update the *dir* value.

3. If the current boundary element P_n is equal to the second border element P_1, and if the previous border element P_{n-1} is equal to P_0, stop. Otherwise repeat step (2).

4. The detected inner border is represented by pixels $P_0 \ldots P_{n-2}$.

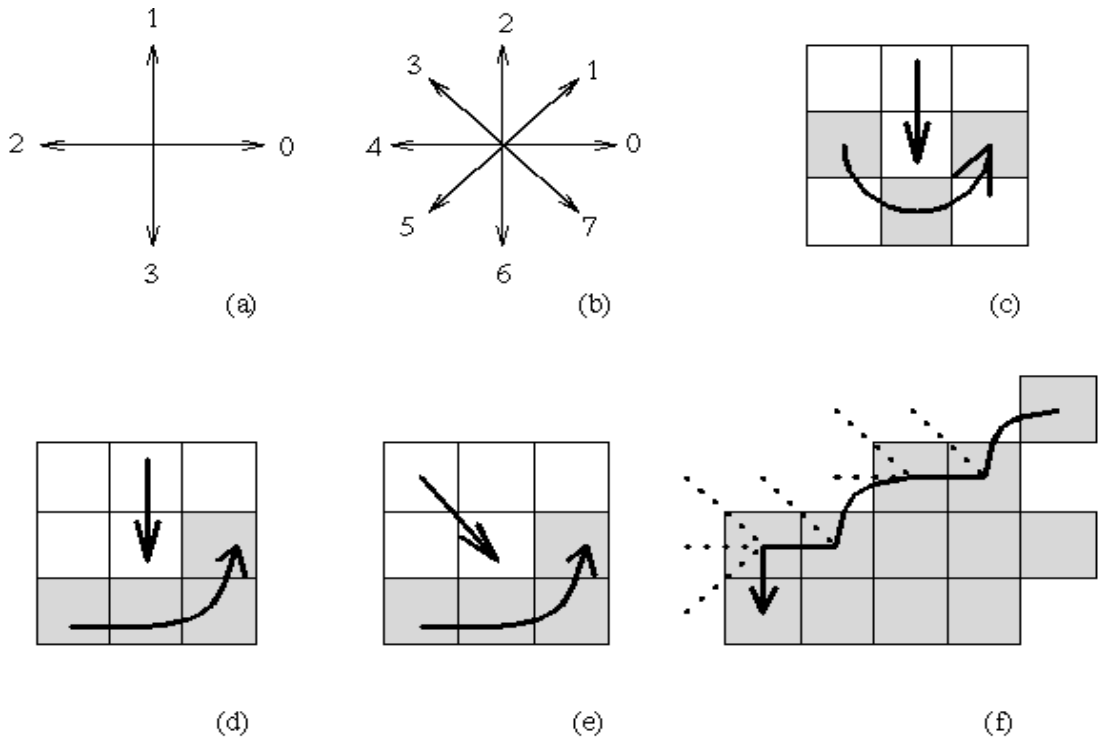


Figure 5.13 *Inner boundary tracing: (a) Direction notation, 4-connectivity, (b) 8-connectivity, (c) pixel neighborhood search sequence in 4-connectivity, (d),(e) search sequence in 8-connectivity, (f) boundary tracing in 8-connectivity (dashed lines show pixels tested during the border tracing).*

- The above Algorithm works for all regions larger than one pixel.
- Looking for the border of a single-pixel region is a trivial problem.
- This algorithm is able to find region borders but does not find borders of region holes.
- To search for hole borders as well, the border must be traced starting in each region or hole border element if this element has never been a member of any border previously traced.
- Note that if objects are of unit width, more conditions must be added.
- If the goal is to detect an **outer region border**, the given algorithm may still be used based on 4-connectivity.
- Note that some border elements may be repeated in the outer border up to three times.

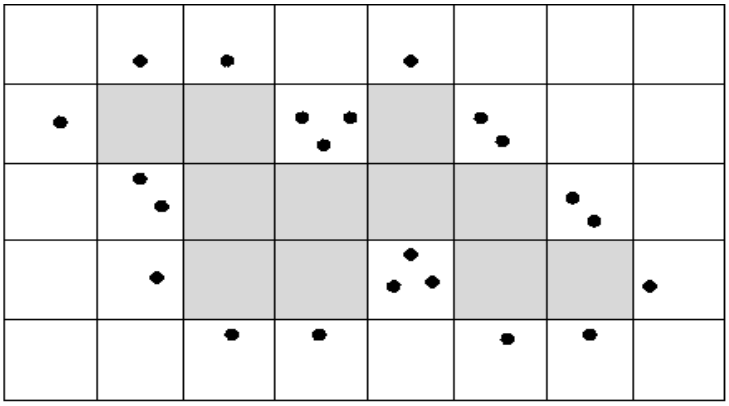
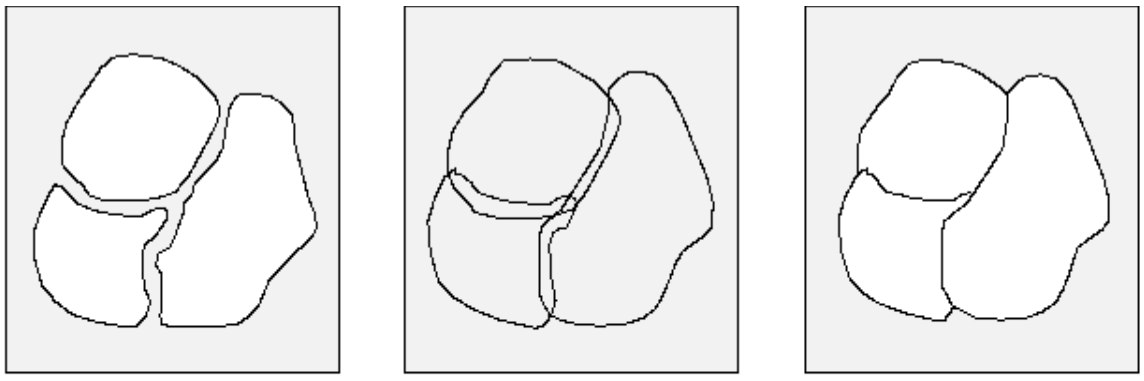


Figure 5.14 *Outer boundary tracing; • denotes outer border elements. Note that some pixels may be listed several times.*

Algorithm 5.7: Outer boundary tracing

1. Trace the inner region boundary in 4-connectivity until done.
2. The outer boundary consists of all non-region pixels that were tested during the search process; if some pixels were tested more than once, they are listed more than once in the outer boundary list.

- The inner border is always part of a region but the outer border never is.
- Therefore, if two regions are adjacent, they never have a common border, which causes difficulties in higher processing levels with region description, region merging, etc.
- The inter-pixel boundary extracted, for instance, from crack edges is common to adjacent borders; nevertheless, its position cannot be specified in pixel co-ordinates.
- Boundary properties better than those of outer borders may be found in **extended** borders.
    - Single common border between adjacent regions.
    - May be specified using standard pixel co-ordinates.
        - The boundary shape is exactly equal to the inter-pixel shape but is shifted one half-pixel down and one half-pixel right.



(a) (b) (c)

Figure 5.15 *Boundary locations for inner, outer, and extended boundary definition: (a) Inner, (b) outer, (c) extended.*

- The extended boundary is defined using 8-neighborhoods
- e.g.P_4(P) denotes the pixel immediately to the left of pixel P
- Four kinds of inner boundary pixels of a region R are defined; if Q denotes pixels outside the region R, then

  LEFT pixel if P_4(P) in Q

  RIGHT pixel if P_0(P) in Q

  UPPER pixel if P_2(P) in Q

  LOWER pixel if P_6(P) in Q
- Let LEFT(R), RIGHT(R), UPPER(R), LOWER(R) represent the corresponding subsets of R.
- The extended boundary EB is defined as a set of points P,P_0,P_6,P_7 satisfying the following conditions:

$$\begin{aligned} EB = \; & \{P : P \in \mathrm{LEFT}(R)\} \cup \{P : P \in \mathrm{UPPER}(R)\} \cup \\ & \{P_6(P) : P \in \mathrm{LOWER}(R)\} \cup \{P_6(P) : P \in \mathrm{LEFT}(R)\} \cup \\ & \{P_0(P) : P \in \mathrm{RIGHT}(R)\} \cup \{P_7(P) : P \in \mathrm{RIGHT}(R)\} \end{aligned} \tag{5.18}$$

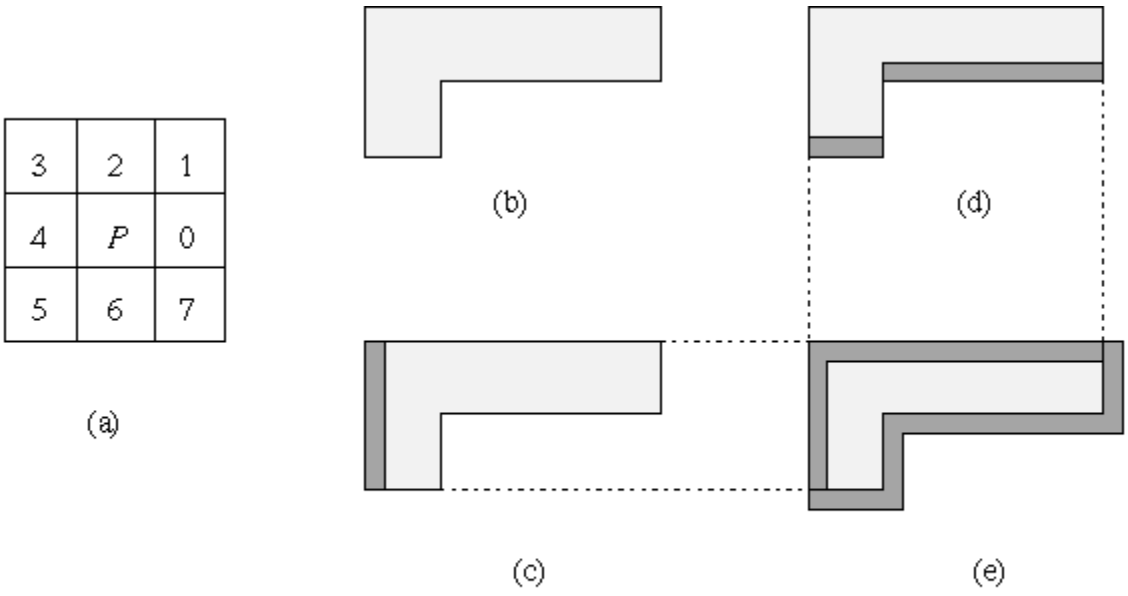


Figure 5.16 *Extended boundary definition: (a) Pixel coding scheme, (b) region R, (c) LEFT(R), (d) LOWER(R), (e) extended boundary.*

- The extended boundary can easily be constructed from the outer boundary.
- Using an intuitive definition of RIGHT, LEFT, UPPER, and LOWER outer boundary points, the extended boundary may be obtained by shifting all the UPPER outer boundary points one pixel down and right, shifting all the LEFT outer boundary points one pixel to the right, and shifting all the RIGHT outer boundary points one pixel down. The LOWER outer boundary point positions remain unchanged.

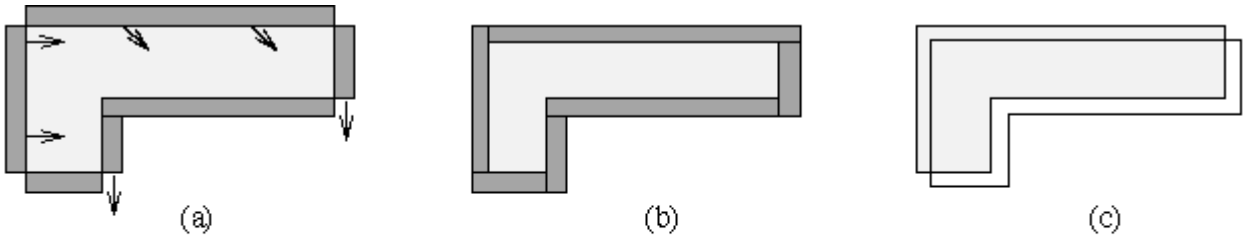


Figure 5.17 *Constructing the extended boundary from outer boundary: (a) Outer boundary, (b) extended boundary construction, (c) extended boundary has the same shape and size as the natural object boundary.*

- A more sophisticated approach is based on detecting common boundary segments between adjacent regions and vertex points in boundary segment connections.

- The detection process is based on a look-up table, which defines all 12 possible situations of the local configuration of 2 x 2 pixel windows, depending on the previous detected direction of boundary, and on the status of window pixels which can be inside or outside a region.

Algorithm 5.8: Extended boundary tracing

1. Define a starting pixel of an extended boundary in a standard way (the first region pixel found in a left-to-right and top-to-bottom line-by-line image search).
2. The first move along the traced boundary from the starting pixel is in direction $dir = 6$ (down), corresponding to the situation (i) in Figure 5.18.
3. Trace the extended boundary using the look-up table in Figure 5.18 until a closed extended border results.

- Note that there is no hole-border tracing included in the algorithm.
- The holes are considered separate regions and therefore the borders between the region and its hole are traced as a border of the hole.

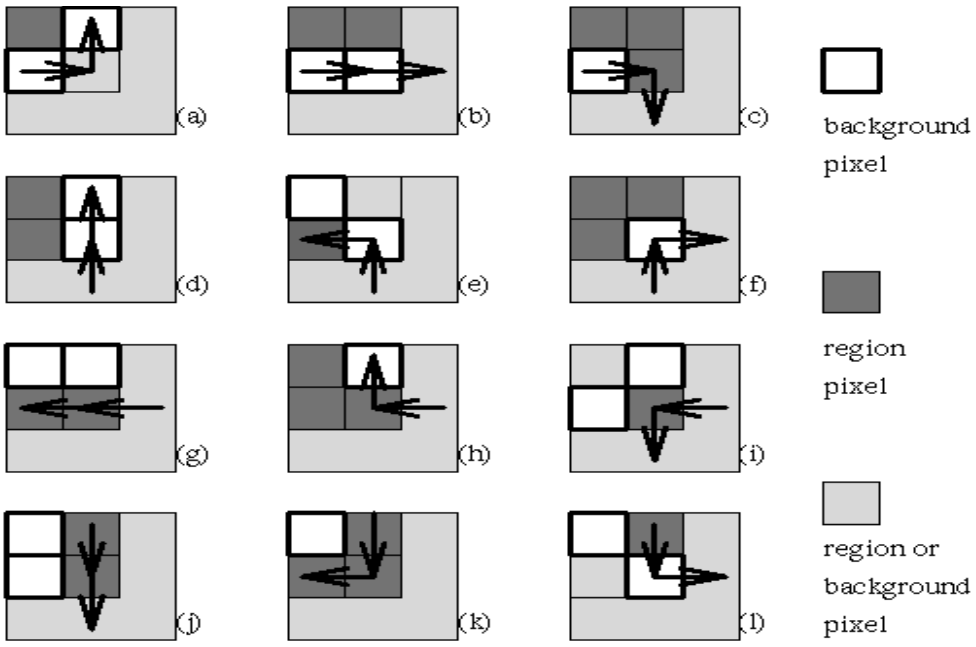


Figure 5.18 ***Look-up table defining all 12 possible situations that can appear during extended border tracing. Current position is in the central pixel. The direction of the next move depends on the local configuration of background and region points, and on the direction of approach to the current pixel. Adapted from [Liow 91].***

- The look-up table approach makes the tracing more efficient than conventional methods and makes parallel implementation possible.
- In addition to extended boundary tracing, it provides a description of each boundary segment in chain code form together with information about vertices.
- This method is very suitable for representing borders in higher level segmentation approaches including methods that integrate edge-based and region-based segmentation results.
- Moreover, in the conventional approaches, each border between two regions must be traced twice. The algorithm can trace each boundary segment only once storing the information about what has already been done in double-linked lists.
- A more difficult situation is encountered if the borders are traced in grey level images where regions have not yet been defined.
- The border is represented by a simple path of high-gradient pixels in the image.
- Border tracing should be started in a pixel with a high probability of being a border element, and then border construction is based on the idea of adding the next elements which are in the most probable direction.

- To find the following border elements, edge gradient magnitudes and directions are usually computed in pixels of probable border continuation.

REGION-BASED SEGMENTATION:

- Edge-based segmentation: borders between regions
- Region-based segmentation: direct construction of regions
- It is easy to construct regions from their borders and it is easy to detect borders of existing regions. Segmentations resulting from edge-based methods and region growing methods are not usually exactly the same. Combination of results may often be a good idea.
- Region growing techniques are generally better in noisy images where edges are extremely difficult to detect.
- Homogeneity of regions is used as the main segmentation criterion in region growing.
- The criteria for homogeneity:
  - gray level
  - color, texture
  - shape
  - model

$$R = \bigcup_{i=1}^{S} R_i \qquad R_i \cap R_j = \emptyset \qquad i \neq j \qquad (5.1)$$

- Regions have already been defined
- Further assumptions:

$$H(R_i) = TRUE \qquad i = 1, 2, ..., S \qquad (5.31)$$

$$H(R_i \cup R_j) = FALSE \quad i \neq j, \quad R_i \ \ adjacent\ to \ \ R_j \qquad (5.32)$$

- Resulting regions of the segmented image must be both homogeneous and maximal.

A. REGION MERGING:

Algorithm 5.16: Region merging (outline)

1. Define some starting method to segment the image into many small regions satisfying condition 5.31.
2. Define a criterion for merging two adjacent regions.
3. Merge all adjacent regions satisfying the merging criterion. If no two regions can be merged maintaining condition 5.31, stop.

- Specific methods differ in the definition of the starting segmentation and in the criterion for merging.
- The result of region merging usually depends on the order in which regions are merged.
- The simplest methods begin merging by starting the segmentation using regions of 2x2, 4x4 or 8x8 pixels.
- Region descriptions are then based on their statistical gray level properties.
- A region description is compared with the description of an adjacent region; if they match, they are merged into a larger region and a new region description is computed. Otherwise regions are marked as non-matching.
- Merging of adjacent regions continues between all neighbors, including newly formed ones. If a region cannot be merged with any of its neighbors, it is marked `final' and the merging process stops when all image regions are so marked.

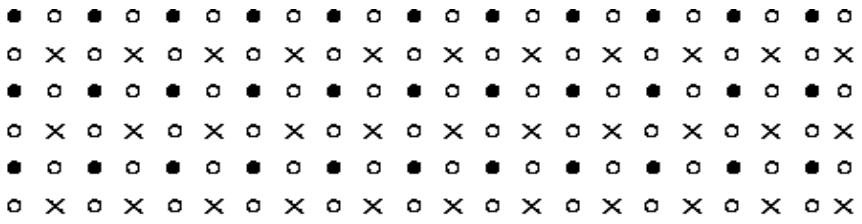


Figure 5.41 *Supergrid data structure: × - image data, ○ - crack edges, ● - unused.*

- The data structure called **supergrid** carries all the necessary information for region merging in 4-adjacency using crack edges.
  - **Merging heuristics:**
    - Two adjacent regions are merged if a significant part of their common boundary consists of weak edges

- ❖ Two adjacent regions are also merged if a significant part of their common boundary consists of weak edges, but in this case not considering the total length of the region borders.

- Of the two given heuristics, the first is more general and the second cannot be used alone because it does not consider the influence of different region sizes.
- Edge significance can be evaluated according to the formula

$$\begin{aligned} v_{ij} &= 0 \quad \text{if } s_{ij} < T_1 \\ &= 1 \quad \text{otherwise} \end{aligned} \tag{5.33}$$

Where v_ij=1 indicates a significant edge, v_ij=0 a weak edge, T_1 is a preset threshold, and s_ij is the crack edge value [s_ij = |f(x_i) - f(x_j)|].

Algorithm 5.17: Region merging via boundary melting

1. Define a starting image segmentation into regions of constant grey level. Construct a supergrid edge data structure in which to store the crack edge information.

2. Remove all weak crack edges from the edge data structure (using equation 5.33 and threshold T_1).

3. Recursively remove common boundaries of adjacent regions R_i, R_j, if

$$\frac{W}{\min(l_i, l_j)} \geq T_2$$

where W is the number of weak edges on the common boundary and l_i, l_j are the perimeter lengths of regions R_i, R_j, and T_2 is another preset threshold.

4. Recursively remove common boundaries of adjacent regions R_i, R_j if

$$\frac{W}{l} \geq T_3 \tag{5.34}$$

or, using a weaker criterion [Ballard and Brown 82]

$$W \geq T_3 \tag{5.35}$$

where l is the length of the common boundary and T_3 is a third threshold.

- The supergrid data structure allows precise work with edges and borders but a big disadvantage of this data structure is that it is not suitable for the representation of regions.
- A good data structure to use can be a planar region adjacency graph.

B. REGION SPLITTING

- Region splitting is the opposite of region merging.
- It begins with the whole image represented as a single region which does not usually satisfy the condition of homogeneity.

- The existing image regions are sequentially split to satisfy all the above given conditions of homogeneity.
- Region splitting does not result in the same segmentation even if the same homogeneity criteria are used.

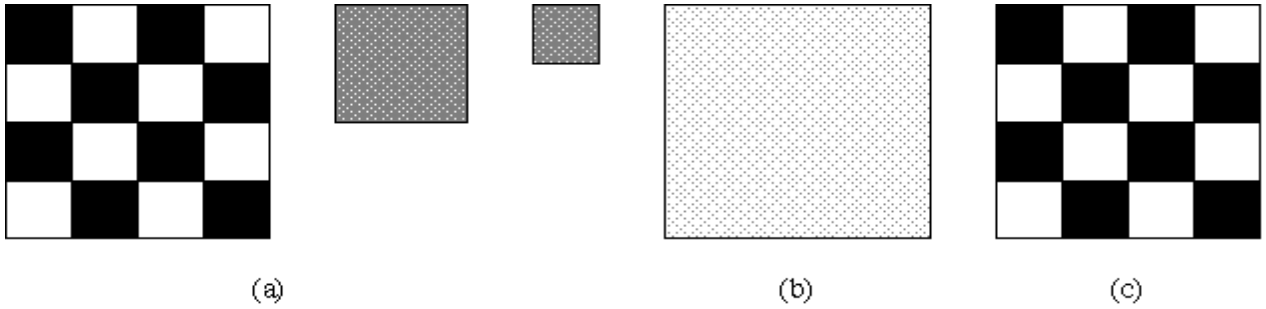


Figure 5.43 *Different segmentations may result from region splitting and region merging approaches: (a) Chessboard image, corresponding pyramid, (b) region splitting segmentation (upper pyramid level is homogeneous, no splitting possible), (c) region merging segmentation (lowest pyramid level consists of regions that are unhomogeneous and cannot be merged).*

- Region splitting methods generally use similar criteria of homogeneity as region merging methods, and only differ in the direction of their application.

C. SPLITTING AND MERGING

- A combination of splitting and merging may result in a method with the advantages of both approaches.
- Split-and-merge approaches work using pyramid image representations; regions are square-shaped and correspond to elements of the appropriate pyramid level.

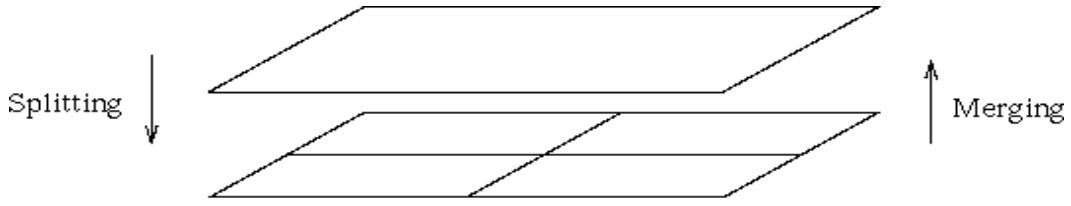


Figure 5.44 *Split-and-merge in a hierarchical data structure.*

- If any region in any pyramid level is not homogeneous (excluding the lowest level), it is split into four subregions -- these are elements of higher resolution at the level below.
- If four regions exist at any pyramid level with approximately the same value of homogeneity measure, they are merged into a single region in an upper pyramid level.

- The segmentation process can be understood as the construction of a segmentation quadtree where each leaf node represents a homogeneous region.
- Splitting and merging corresponds to removing or building parts of the segmentation quadtree.
- Split-and-merge methods usually store the adjacency information in region adjacency graphs (or similar data structures).
- Using segmentation trees, in which regions do not have to be contiguous, is both implementationally and computationally easier.
- An unpleasant drawback of segmentation quadtrees is the square region shape assumption
    - merging of regions which are not part of the same branch of the segmentation tree
- Because both split-and-merge processing options are available, the starting segmentation does not have to satisfy any of the homogeneity conditions.

Algorithm 5.18: Split and merge

1. Define an initial segmentation into regions, a homogeneity criterion, and a pyramid data structure.

2. If any region R in the pyramid data structure is not homogeneous ($H(R) = FALSE$), split it into 4 child-regions; if any 4 regions with the same parent can be merged into a single homogeneous region, merge them. If no region can be split or merged, go to step (3).

3. If there are any two adjacent regions R_i, R_j (even if they are in different pyramid levels or do not have the same parent) that can be merged into a homogeneous region, merge them.

4. Merge small regions with the most similar adjacent region if it is necessary to remove small-size regions.

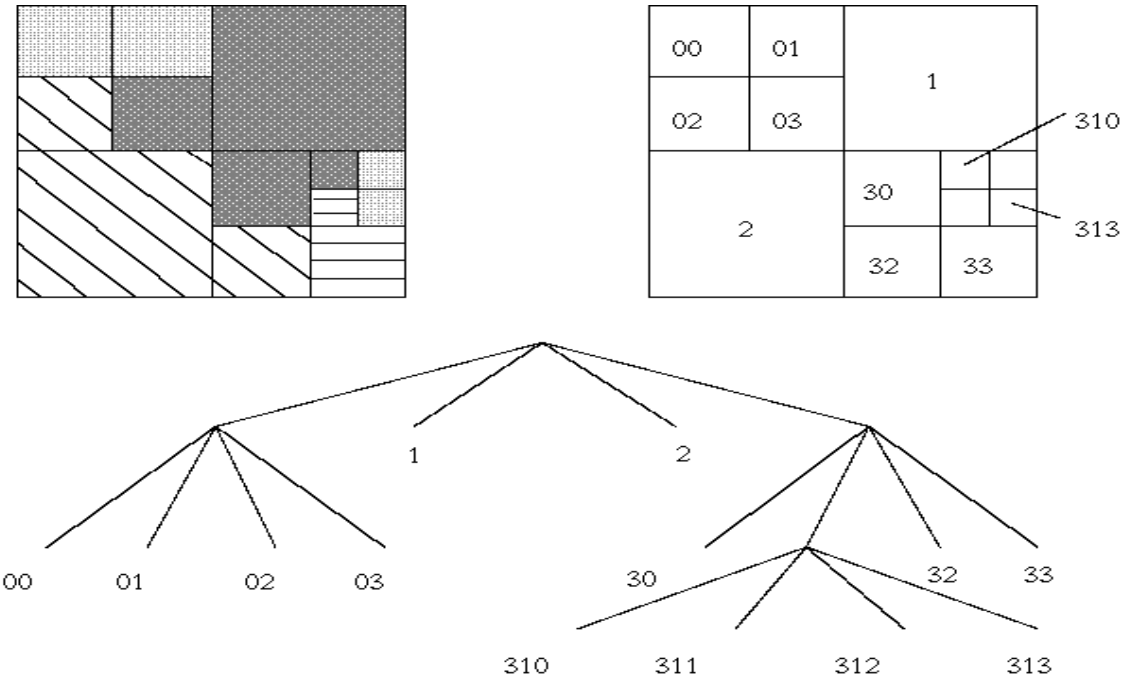


Figure 5.45 *Segmentation quadtree.*

- A pyramid data structure with overlapping regions is an interesting modification of this method.
- Each region has four potential parent elements in the upper pyramid level and sixteen possible child elements in the lower pyramid level.

- Segmentation tree generation begins in the lowest pyramid level. Properties of each region are compared with properties of each of its potential parents and the segmentation branch is linked to the most similar of them.
- After construction of the tree is complete, all the homogeneity values of all the elements in the pyramid data structure are recomputed to be based on child-region properties only.
- This recomputed pyramid data structure is used to generate a new segmentation tree, beginning again at the lowest level.
- The pyramid updating process and new segmentation tree generation is repeated until no significant segmentation changes can be detected between steps.
- The highest level of the segmentation tree must correspond to the expected number of image regions and the pyramid height defines the maximum number of segmentation branches.
- If the number of regions in an image is less than 2^n, some regions can be represented by more than one element in the highest pyramid level.
- Considerably lower memory requirements can be found in a **single-pass split-and-merge** segmentation.
- A local splitting pattern is detected in each 2x2 pixel image block and regions are merged in overlapping blocks of the same size.

Algorithm 5.19: Split and link to the segmentation tree

1. Define a pyramid data structure with overlapping regions. Evaluate the starting region description.

2. Build a segmentation tree starting with leaves. Link each node of the tree to that one of the four possible parents to which it has the most similar region properties. Build the whole segmentation tree. If there is no link to an element in the higher pyramid level, assign the value zero to this element.

3. Update the pyramid data structure; each element must be assigned the average of the values of all its existing children.

4. Repeat steps (2) and (3) until no significant segmentation changes appear between iterations (a small number of iterations is usually sufficient).

Algorithm 5.20: Single-pass split-and-merge

1. Search an entire image line by line except the last column and last line. Perform the following steps for each pixel.

2. Find a splitting pattern for a 2×2 pixel block.

3. If a mismatch between assigned labels and splitting patterns in overlapping blocks is found, try to change the assigned labels of these blocks to remove the mismatch (discussed below).

4. Assign labels to unassigned pixels to match a splitting pattern of the block.

5. Remove small regions if necessary.

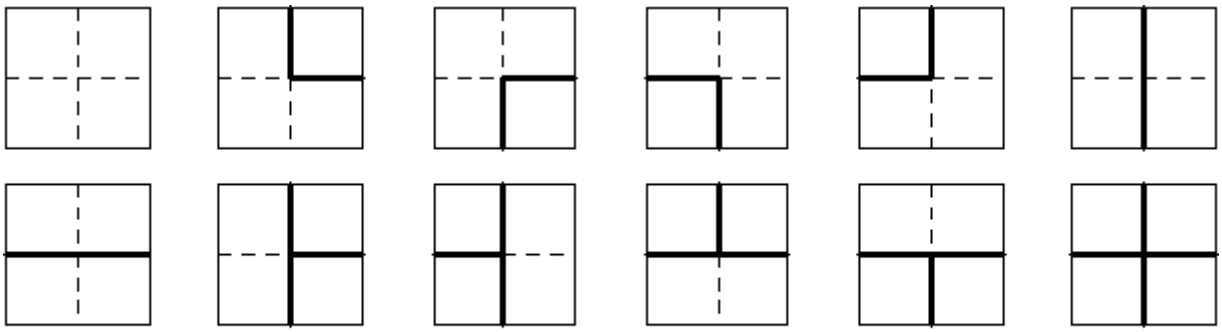


Figure 5.46 *Splitting of 2×2 image blocks, all 12 possible cases.*

- The image blocks overlap during the image search.
- Except for locations at the image borders, three of the four pixels have been assigned a label in previous search locations, but these labels do not necessarily match the splitting pattern found in the processed block.
- If a mismatch is detected in step 3 of the algorithm, it is necessary to resolve possibilities of merging regions that were considered separate so far - to assign the same label to two regions previously labeled differently.
- Two regions R_1 and R_2 are merged into a region R_3 if

$$H(R_1 \cup R_2) = TRUE \qquad (5.36)$$

$$|m_1 - m_2| < T \qquad (5.37)$$

- Where m_1 and m_2 are the mean gray level values in regions R_1 and R_2, and T is some appropriate threshold.
- If region merging is not allowed, regions keep their previous labels.
- If larger blocks are used, more complex image properties can be included in the homogeneity criteria (even if these larger blocks are divided into 2x2 sub-blocks to determine the splitting pattern).

D. WATERSHED SEGMENTATION:

- The concepts of **watersheds** and **catchment basins** are well known in topography.
- Watershed lines divide individual catchment basins.
- The North American Continental Divide is a textbook example of a watershed line with catchment basins formed by the Atlantic and Pacific Oceans.
- Image data may be interpreted as a topographic surface where the gradient image gray-levels represent altitudes.

- Region edges correspond to high watersheds and low-gradient region interiors correspond to catchment basins.
- Catchment basins of the topographic surface are homogeneous in the sense that all pixels belonging to the same catchment basin are connected with the basin's region of minimum altitude (gray-level) by a *simple path* of pixels that have monotonically decreasing altitude (gray-level) along the path.
- Such catchment basins then represent the regions of the segmented image.

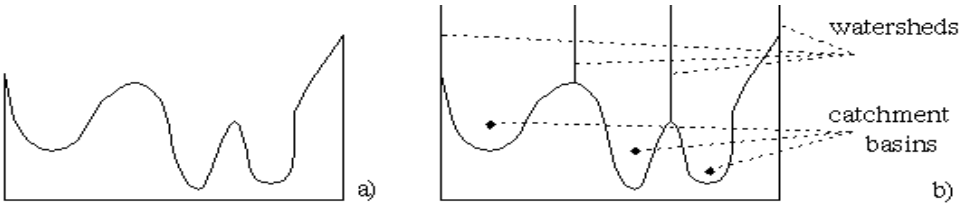


Figure 5.47 *One-dimensional example of watershed segmentation. (a) Gray level profile of image data. (b) Watershed segmentation – local minima of gray level (altitude) yield catchment basins, local maxima define the watershed lines.*

- Concept of watersheds and catchment basins is quite straightforward.
- Early watershed methods resulted in either slow or inaccurate execution.

- Most of the existing algorithms start with extraction of potential watershed line pixels using a local 3 x 3 operation, which are then connected into geomorphological networks in subsequent steps. Due to the local character of the first step, these approaches are often inaccurate.
- A watershed transformation was also introduced in the context of mathematical morphology - computationally demanding and therefore time consuming.
- Two basic approaches to watershed image segmentation.
    - The first one starts with finding a *downstream* path from each pixel of the image to a local minimum of image surface altitude.
    - A catchment basin is then defined as the set of pixels for which their respective downstream paths all end up in the same altitude minimum.
    - While the downstream paths are easy to determine for continuous altitude surfaces by calculating the local gradients, no rules exist to define the downstream paths uniquely for digital surfaces.
    - The second approach is essentially dual to the first one; instead of identifying the downstream paths, the catchment basins fill from the bottom.

- Imagine that there is a hole in each local minimum, and that the topographic surface is immersed in water - water starts filling all catchment basins, minima of which are under the water level. If two catchment basins would merge as a result of further immersion, a dam is built all the way to the highest surface altitude and the dam represents the watershed line.
- An efficient algorithm is based on *sorting* the pixels in increasing order of their gray values, followed by a *flooding* step consisting of a fast breadth-first scanning of all pixels in the order of their gray-levels.
- During the sorting step, a brightness histogram is computed. Simultaneously, a list of pointers to pixels of gray-level *h* is created and associated with each histogram gray-level to enable direct access to all pixels of any gray-level.
- Information about the image pixel sorting is used extensively in the flooding step.
- Suppose the flooding has been completed up to a level (gray-level, altitude) *k*. Then every pixel having gray-level less than or equal to *k* has already been assigned a unique catchment basin label.
- Next, pixels having gray-level $k+1$ must be processed; all such pixels can be found in the list that was prepared in the sorting step - consequently, all these pixels can be accessed directly.
- A pixel having gray-level $k+1$ may belong to a catchment basin labeled *l* ("el") if at least one of its neighbors already carries this label.
- Pixels that represent potential catchment basin members are put in a first-in first-out queue and await further processing.
- Geodesic influence zones are computed for all hitherto determined catchment basins.
- A geodesic influence zone of a catchment basin *l_i* is the locus of non-labeled image pixels of gray-level $k+1$ that are contiguous with the catchment basin *l_i* (contiguous within the region of pixels of gray-level $k+1$) for which their distance to *l_i* is smaller than their distance to any other catchment basin *l_j*.

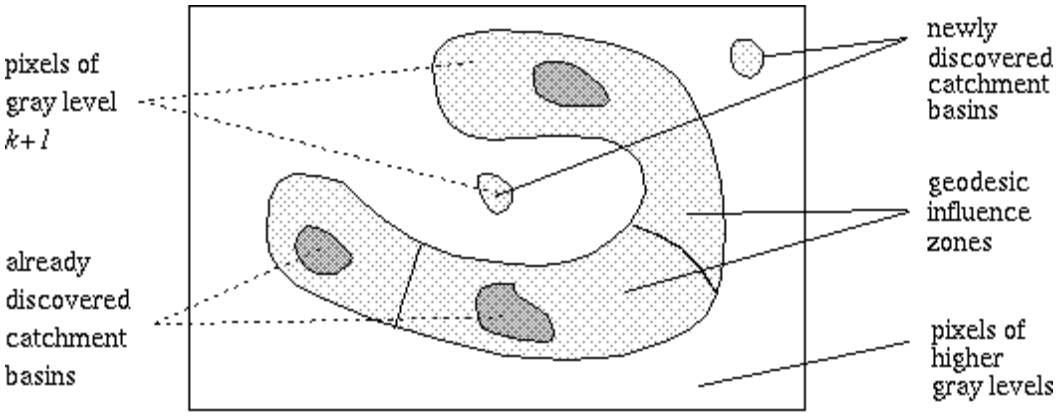


Figure 5.48 *Geodesic influence zones of catchment basins.*

- All pixels with gray-level k+1 that belong to the influence zone of a catchment basin labeled l are also labeled with the label l, thus causing the catchment basin to grow.
- The pixels from the queue are processed sequentially, and all pixels from the queue that cannot be assigned an existing label represent newly discovered catchment basins and are marked with new and unique labels.
- Example of watershed segmentation.
- Raw watershed segmentation produces a severely oversegmented image with hundreds or thousands of catchment basins.
- To overcome this problem, region markers and other approaches have been suggested to generate good segmentation.

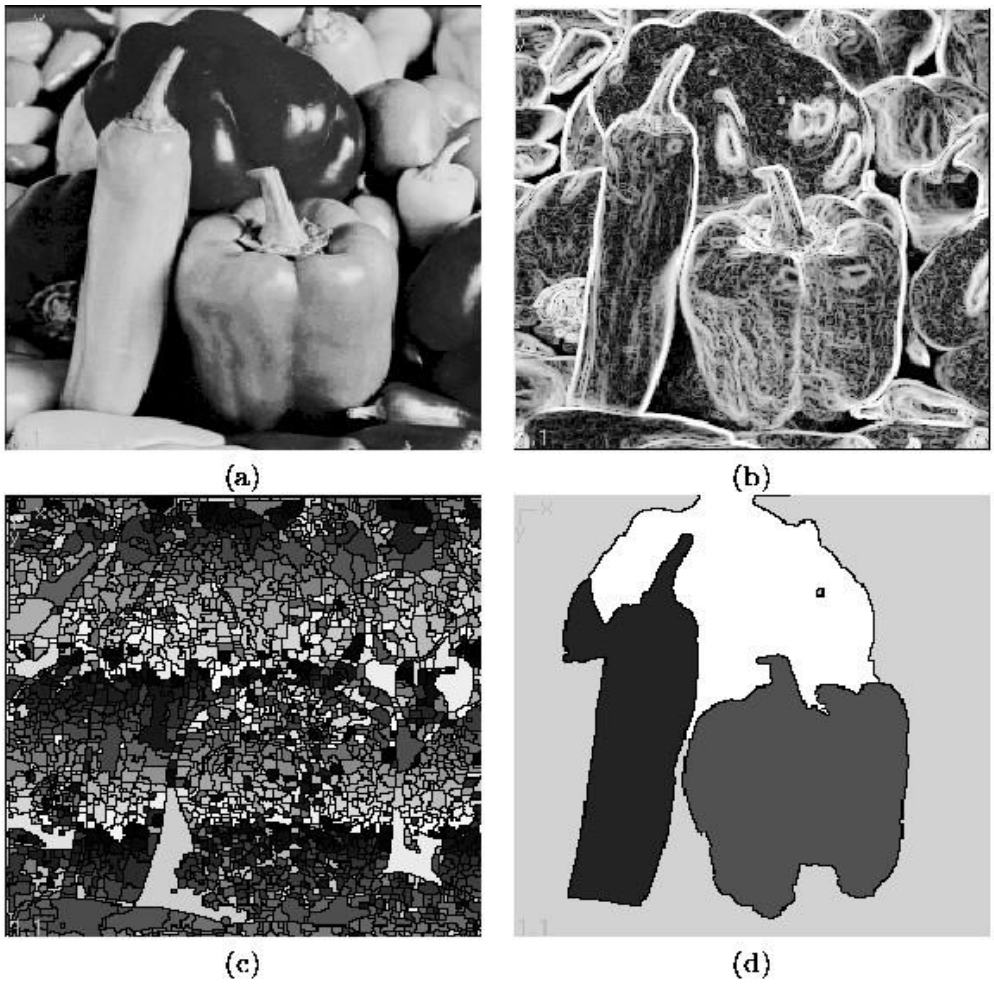


Figure 5.51: *Watershed segmentation: (a) original; (b) gradient image, 3×3 Sobel edge detection, histogram equalized; (c) raw watershed segmentation; (d) watershed segmentation using region markers to control oversegmentation. Courtesy W. Higgins, Penn State University.*

- While this method would work well in the continuous space with the watershed lines accurately dividing the adjacent catchment basins, the watersheds in images with large plateaus may be quite thick in discrete spaces:

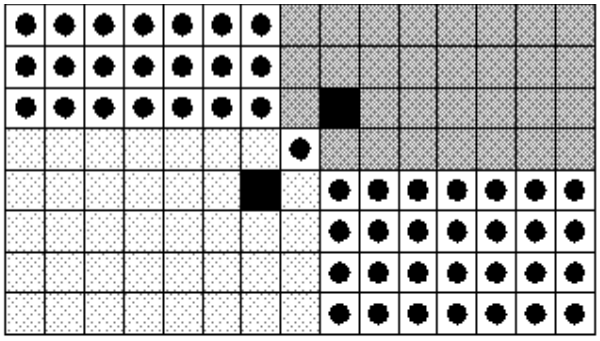


Figure 5.50 *Thick watershed lines may result in gray-level plateaux. Earlier identified catchment basins are marked as black pixels, and new catchment basin additions resulting from this processing step are shown in the two levels of gray. The thick watersheds are marked with •. To avoid thick watersheds, specialized rules must be developed.*

E. REGION GROWING POST-PROCESSING:

- Region growing often results in under growing or overgrowing as a result of non-optimal parameter setting.
- A variety of post-processors has been developed.
- Some of them combine segmentation information obtained from region growing and edge-based segmentation.
- Simpler post-processors are based on general heuristics and decrease the number of small regions in the segmented image that cannot be merged with any adjacent region according to the originally applied homogeneity criteria.

Algorithm 5.21: Removal of small image regions

1. Search for the smallest image region R_{min}.
2. Find the region R most similar to R_{min}, according to the homogeneity criteria used. Merge R and R_{min}.
3. Repeat steps (1) and (2) until all regions smaller than a preselected size are removed from the image.

- This algorithm will execute much faster if all regions smaller than a preselected size are merged with their neighbors without having to order them by size.

MATCHING:

- Matching is another basic approach to segmentation that can be used to locate known objects in an image, to search for specific patterns, etc.
- The best match is based on some criterion of optimality which depends on object properties and object relations.

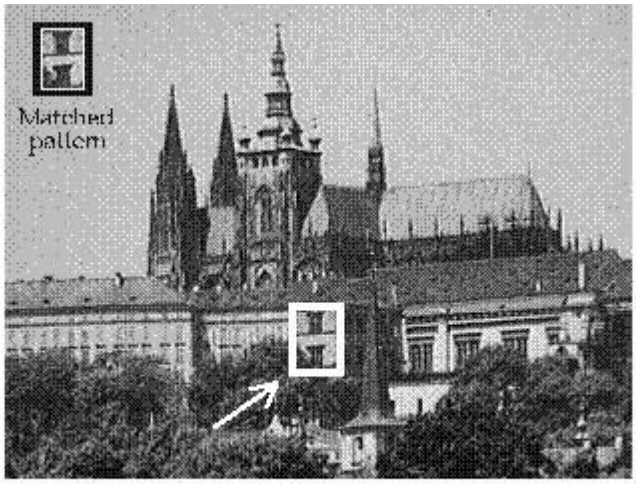


Figure 5.53 *Segmentation by matching; matched pattern and location of the best match.*

- Matched patterns can be very small, or they can represent whole objects of interest.
- While matching is often based on directly comparing gray-level properties of image subregions, it can be equally well performed using image-derived features or higher-level image descriptors.
- In such cases, the matching may become invariant to image transforms.
- Criteria of optimality can compute anything from simple correlations up to complex approaches of graph matching.

A. MATCHING CRITERIA:

- Exact copy of the pattern of interest cannot be expected in the processed image - some part of the pattern is usually corrupted in real images by noise, geometric distortion, occlusion, etc.
- Search for locations of maximum match is appropriate.

Algorithm 5.24: Match-based segmentation

1. Evaluate a match criterion for each location and rotation of the pattern in the image.
2. Local maxima of this criterion exceeding a preset threshold represent pattern locations in the image.

- Matching criteria can be defined in many ways; in particular, correlation between a pattern and the searched image data is a general matching criterion.
- Let f be an image to process, h be a pattern to search for, and V be the set of all image pixels in h.
- Possible matching optimality criteria describing a match between f and h located at a position (u,v):

$$C_1(u,v) = \frac{1}{\max_{(i,j)\in V} |f(i+u, j+v) - h(i,j)| + 1} \qquad (5.38)$$

$$C_2(u,v) = \frac{1}{\sum_{(i,j)\in V} |f(i+u, j+v) - h(i,j)| + 1} \qquad (5.39)$$

$$C_3(u,v) = \frac{1}{\sum_{(i,j)\in V} [f(i+u, j+v) - h(i,j)]^2 + 1} \qquad (5.40)$$

- The "1" added to each denomiator to prevents dividing by zero for a perfect match.
- The cost is evaluated at each (u,v) pixel location in the image being processed.
- Possible implementation decissions include whether the pattern is only computed entirely within the image or partial pattern positions when the pattern crosses image borders.
- A simple example of the C_3 optimality criterion values is given:

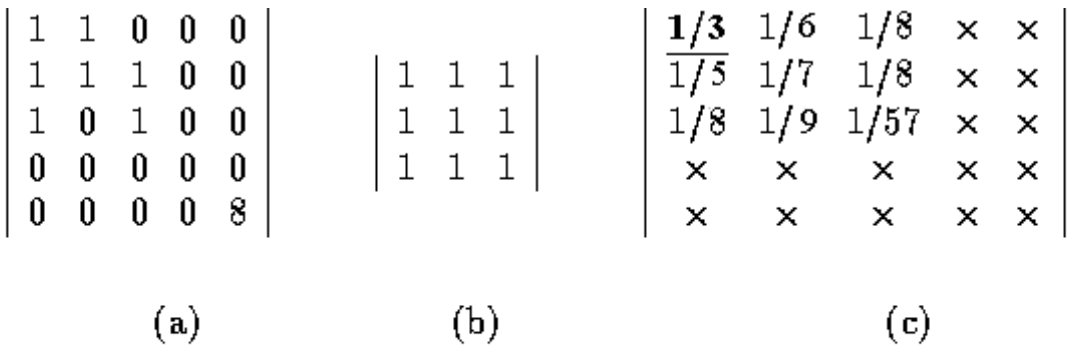


Figure 5.54 *Optimality matching criterion evaluation: (a) Image data, (b) matched pattern, (c) optimality criterion values, the best match underlined.*

- If a fast, effective Fourier transform algorithm is available, the convolution theorem can be used to evaluate matching.
- The correlation between a pattern h and image f can be determined by first taking the product of the Fourier transform F of the image f and the complex conjugate of the Fourier transform H^# of the pattern h and then applying the inverse transform.
- To compute the product of Fourier transforms, F and H^# must be of the same size; if a pattern size is smaller, zero-valued lines and columns can be added to inflate it to the appropriate size.
- Sometimes, it may be better to add non-zero numbers, for example, the average gray level of processed images can serve the purpose well.

B. CONTROL STRATEGIES OF MATCHING:

- Match-based segmentation localizes all image positions at which close copies of the searched pattern are located.
- These copies must match the pattern in size and rotation, and the geometric distortion must be small.
- To adapt match-based methods to detect patterns that are rotated, enlarged, and/or reduced, it would be necessary to consider patterns of all possible sizes and rotations.
- Another option is to use just one pattern and match an image with all possible geometric transforms of this pattern, and this may work well if some information about the probable geometric distortion is available.
- Note that there is no difference in principle between these approaches.

- Matching can be used even if an infinite number of transformations are allowed. Let us suppose a pattern consists of parts, these parts being connected by rubber links.
- Even if a complete match of the whole pattern within an image may be impossible, good matches can often be found between pattern parts and image parts.
- Good matching locations may not be found in the correct relative positions, and to achieve a better match, the rubber connections between pattern parts must be either pushed or pulled.
- Match-based segmentation is time consuming even in the simplest cases with no geometric transformations, but the process can be made faster if a good operation sequence is found.
- The sequence of match tests must be data driven.
    - Fast testing of image locations with a high probability of match may be the first step, and then it is not necessary to test all possible pattern locations.
    - Another speed improvement can be realized if a mismatch can be detected before all the corresponding pixels have been tested.
    - The correlation changes slowly around the best matching location ... matching can be tested at lower resolution first, looking for an exact match in the neighborhood of good low-resolution matches only.
- The mismatch must be detected as soon as possible since mismatches are found much more often than matches.
- Considering the matching formulae given above, testing in a specified position must stop when the value in the denominator (measure of mismatch) exceeds some preset threshold.
- This implies that it is better to begin the correlation test in pixels with a high probability of mismatch in order to get a steep growth in the mismatch criterion.
- This criterion growth will be faster than that produced by an arbitrary pixel order computation.

SHAPE REPRESENTATION AND DESCRIPTION:

- Defining the shape of an object can prove to be very difficult. Shape is usually represented verbally or in figures.
- There is no generally accepted methodology of shape description. Further, it is not known what in shape is important.
- Current approaches have both positive and negative attributes; computer graphics or mathematics use effective shape representations which are unusable in shape recognition and vice versa.
- In spite of this, it is possible to find features common to most shape description approaches.
- Common shape description methods can be characterized from different points of view

- Input representation form: Object description can be based on boundaries or on more complex knowledge of whole regions
- Object reconstruction ability: That is, whether an object's shape can or cannot be reconstructed from the description.
- Incomplete shape recognition ability: That is, to what extent an object's shape can be recognized from the description if objects are occluded and only partial shape information is available.
- Local/global description character: Global descriptors can only be used if complete object data are available for analysis. Local descriptors describe local object properties using partial information about the objects. Thus, local descriptors can be used for description of occluded objects.
- Mathematical and heuristic techniques: A typical mathematical technique is shape description based on the Fourier transform. A representative heuristic method may be elongatedness.
- A robustness of description to translation, rotation, and scale transformations: Shape description properties in different resolutions.

- Sensitivity to scale is even more serious if a shape description is derived, because shape may change substantially with image resolution.
- Therefore, shape has been studied in multiple resolutions which again causes difficulties with matching corresponding shape representations from different resolutions.
- Moreover, the conventional shape descriptions change discontinuously.
- A scale-space approach aims to obtain continuous shape descriptions if the resolution changes continuously.

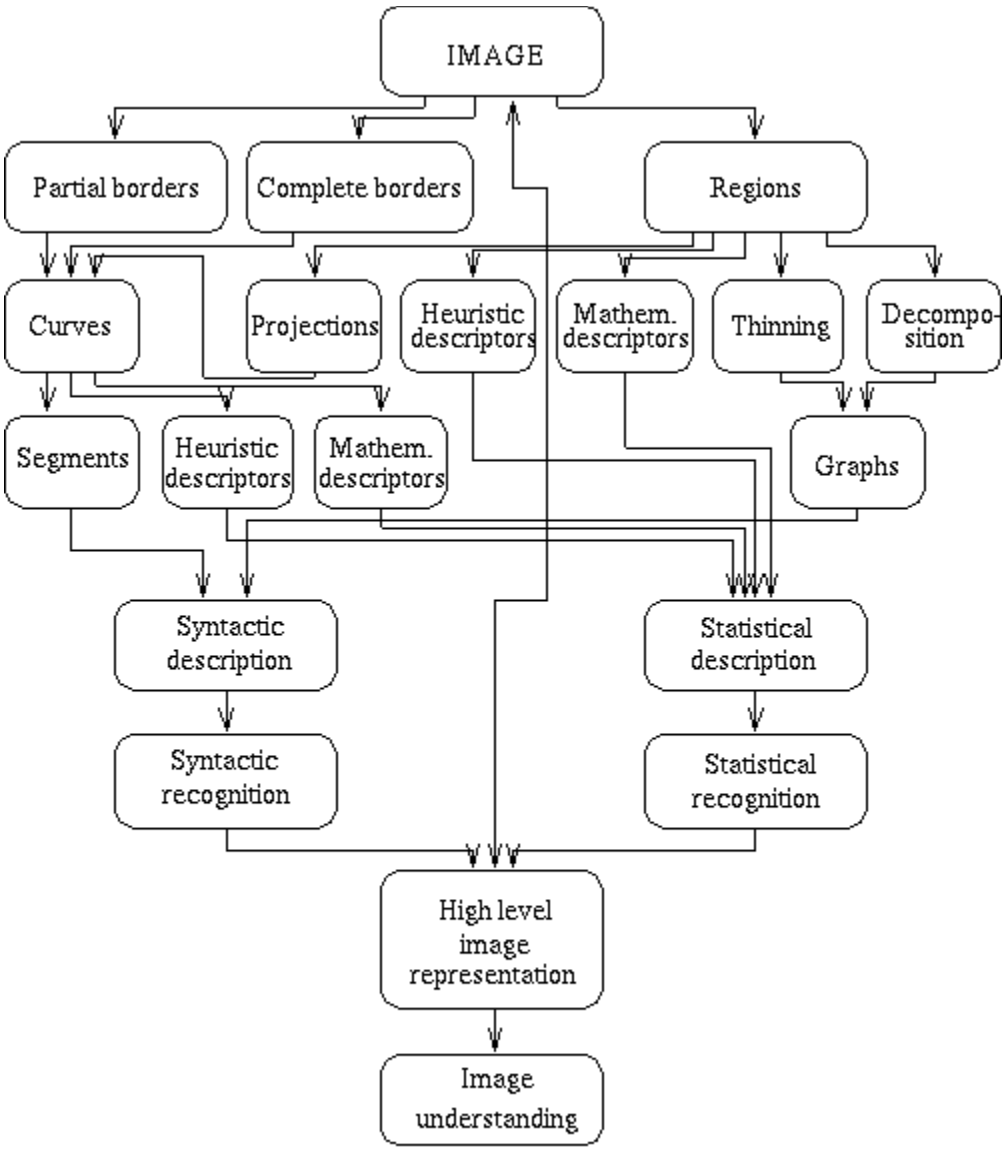


Figure 6.1 *Image analysis and understanding methods.*

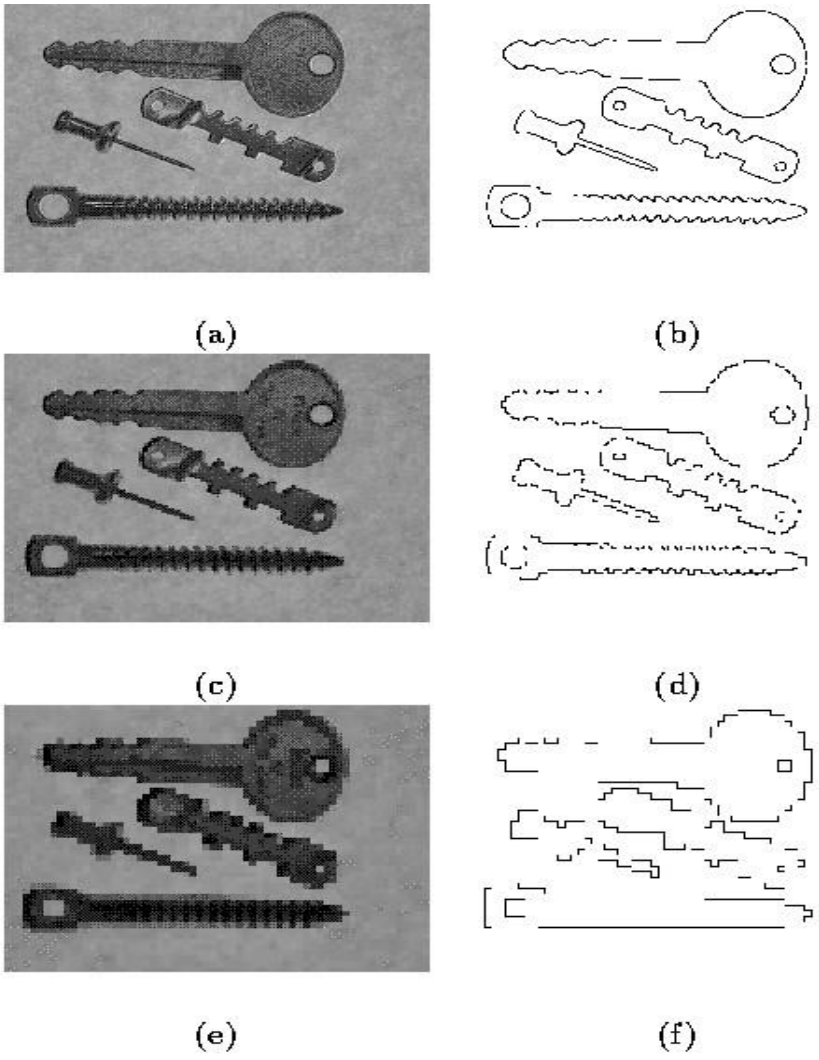


Figure 6.2 *(a) Original image 640 × 480, (b) contours of a, (c) original image 160 × 120 (d) contours of c, (e) original image 64 × 48, (f) contours of e.*

- In many tasks, it is important to represent classes of shapes properly, e.g. shape classes of apples, oranges, pears, bananas, etc.
- The **shape classes** should represent the generic shapes of the objects belonging to the same classes well. Obviously, shape classes should emphasize shape differences among classes while the influence of shape variations within classes should not be reflected in the class description.
- Despite the fact that we are dealing with two-dimensional shape and its description, our world is three-dimensional and the same objects, if seen from different angles (or changing position/orientation in space), may form very different 2D projections.

- The ideal case would be to have a universal shape descriptor capable of overcoming these changes -- to design projection-invariant descriptors.
- Consider an object with planar faces and imagine how many very different 2D shapes may result from a given face if the position and 3D orientation of this simple object changes with respect to an observer. In some special cases, like circles which transform to ellipses, or planar polygons, projectively invariant features (**invariants**} can be found.
- Object **occlusion** is another hard problem in shape recognition. However, the situation is easier here (if pure occlusion is considered, not combined with orientation variations yielding changes in 2D projections as discussed above), since visible parts of objects may be used for description.
- Here, the shape descriptor choice must be based on its ability to describe local object properties -- if the descriptor only gives a global object description, such a description is useless if only a part of an object is visible.

A. REGION IDENTIFICATION:

- Region identification is necessary for region description. One of the many methods for region identification is to label each region (or each boundary) with a unique (integer) number; such identification is called **labeling** or **coloring**, also **connected component labeling.** Goal of segmentation was to achieve complete segmentation, now, the regions must be labeled.

$$R_b^C = \bigcup_{i=1, i\neq b}^{m} R_i$$

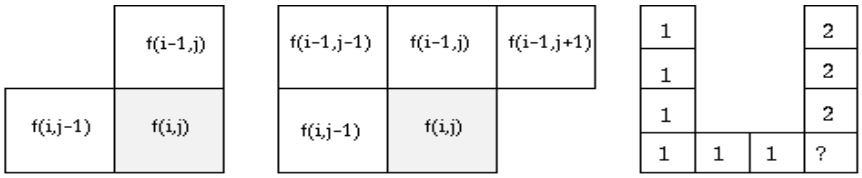


Figure 6.3 *Masks for region identification: (a) In 4-connectivity, (b) in 8-connectivity, (c) label collision.*

- **Label collision** is a very common occurrence -- examples of image shapes experiencing this are U-shaped objects, mirrored E objects, etc.
- The equivalence table is a list of all label pairs present in an image; all equivalent labels are replaced by a unique label in the second step.
- The algorithm is basically the same in 4-connectivity and 8-connectivity, the only difference being in the neighborhood mask shape.

Algorithm 6.1: 4-neighborhood and 8-neighborhood region identification

1. First pass: Search the entire image R row by row and assign a non-zero value v to each non-zero pixel $R(i,j)$. The value v is chosen according to the labels of the pixel's neighbors where the property *neighboring* is defined by Figure 6.3. ('neighbors' outside the image R are not considered),

   - If all the neighbors are background pixels (with pixel value zero), $R(i,j)$ is assigned a new (and as yet) unused label.
   - If there is just one neighboring pixel with a non-zero label, assign this label to the pixel $R(i,j)$.
   - If there is more than one non-zero pixel among the neighbors, assign the label of any one to the labelled pixel. If the labels of any of the neighbors differ (*label collision*) store the label pair as being equivalent. Equivalence pairs are stored in a separate data structure – an equivalence table.

2. Second pass: All of the region-pixels were labelled during the first pass but some regions have pixels with different labels (due to label collisions). The whole image is scanned again, and pixels re-labelled using the equivalence table information (for example, with the lowest value in an equivalence class).

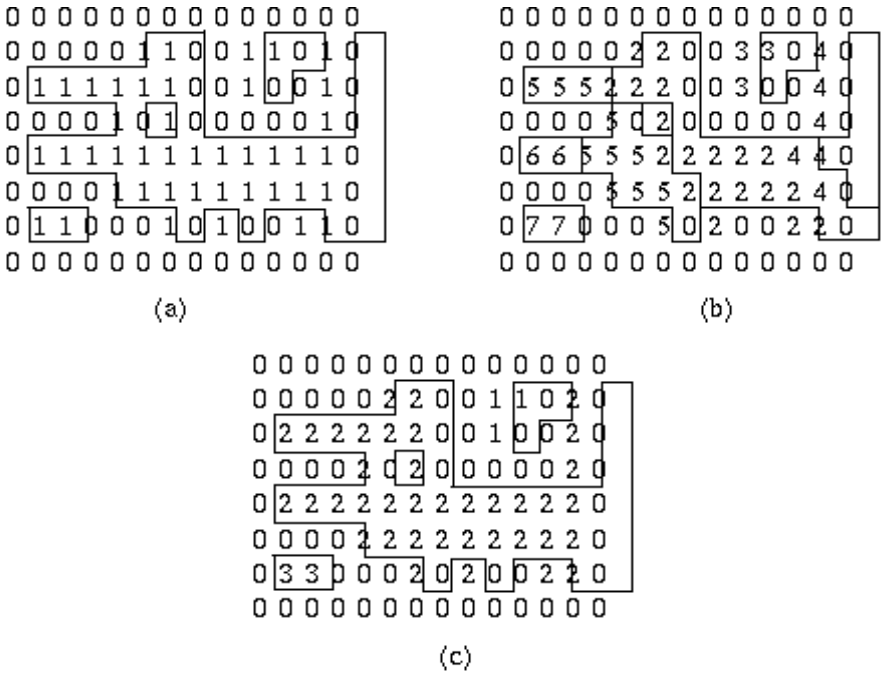


Figure 6.4 *Object identification in 8-connectivity: (a),(b),(c) Algorithm steps. Equivalence table after step (b): 2-5, 5-6, 2-4.*

Algorithm 6.2: Region identification in run-length encoded data

1. First pass: Use a new label for each continuous run in the first image row that is not part of the background.

2. For the second and subsequent rows, compare positions of runs. If a run in a row does not neighbor (in the 4- or 8- sense) any run in the previous row, assign a new label. If a run neighbors precisely one run in the previous row, assign its label to the new run. If the new run neighbors more than one run in the previous row, a label collision has occurred. Collision information is stored in an equivalence table, and the new run is labelled using the label of any one of its neighbors.

3. Second pass: Search the image row by row and re-label the image according to the equivalence table information.

Algorithm 6.3: Quadtree region identification

1. First pass: Search quadtree nodes in a given order – e.g. beginning from the root and in NW, NE, SW, SE directions. Whenever an unlabelled non-zero leaf node is entered, a new label is assigned to it. Then search for neighboring leaf nodes in the E and S directions (plus SE in 8-connectivity). If those leaves are non-zero and have not yet been labelled, assign the label of the node from which the search started. If the neighboring leaf node has already been labelled, store the collision information in an equivalence table.
2. Repeat step (1) until the whole tree has been searched.
3. Second pass: Re-label the leaf nodes of the quadtree according to the equivalence table.

CONTOUR-BASED SHAPE REPRESENTATION AND DESCRIPTION:

- Region borders must be expressed in some mathematical form.

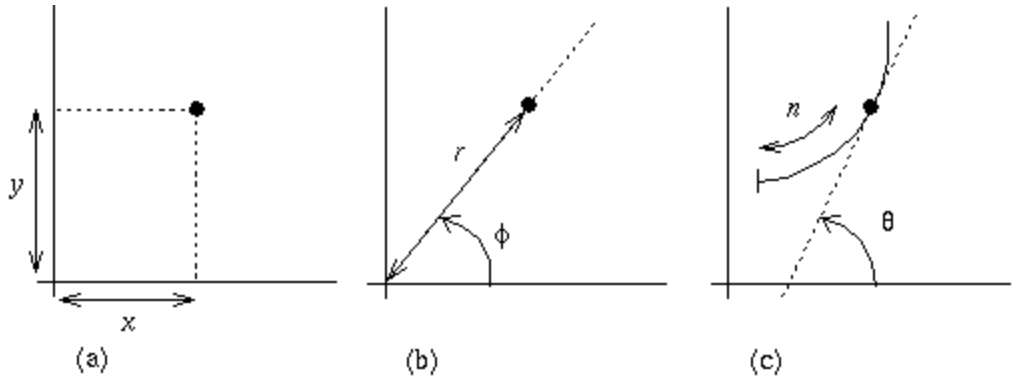


Figure 6.5 *Co-ordinate systems: (a) Rectangular (Cartesian), (b) polar, (c) tangential.*

A. CHAIN CODES:

- Chain codes describe an object by a sequence of unit-size line segments with a given orientation.
- The first element of such a sequence must bear information about its position to permit the region to be reconstructed.

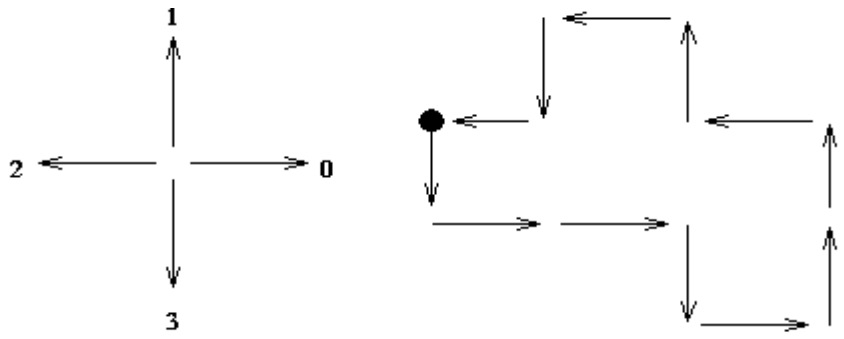


Figure 6.6 *Chain code in 4-connectivity, and its derivative. Code: 3, 0, 0, 3, 0, 1, 1, 2, 1, 2, 3, 2, derivative: 1, 0, 3, 1, 1, 0, 1, 3, 1, 1, 3, 1.*

- If the chain code is used for matching it must be independent of the choice of the first border pixel in the sequence. One possibility for normalizing the chain code is to find the pixel in the border sequence which results in the minimum integer number if the description chain is interpreted as a base four number -- that pixel is then used as the starting pixel.
    - A *mod* 4 or *mod* 8 differences are called a chain code **derivative**.

B. SIMPLE GEOMETRIC BORDER REPRESENTATION:

- The following descriptors are mostly based on geometric properties of described regions. Because of the discrete character of digital images, all of them are sensitive to image resolution.
    - ✓ *Boundary length*
    - ✓ *Curvature*

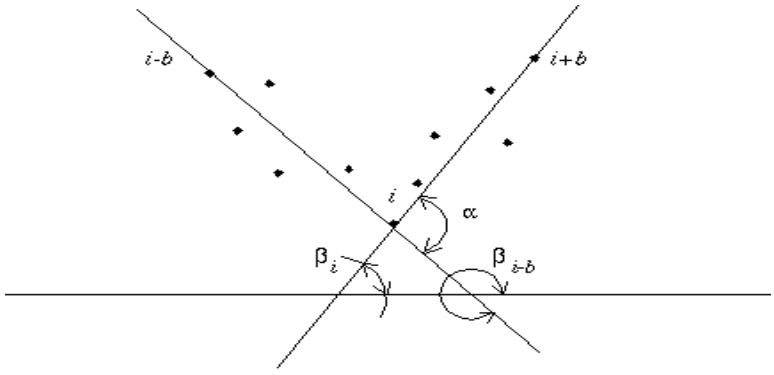


Figure 6.7 *Curvature.*

- ***Bending energy***

$$BE = \frac{1}{L}\sum_{k=1}^{L} c^2(k) \qquad (6.1)$$

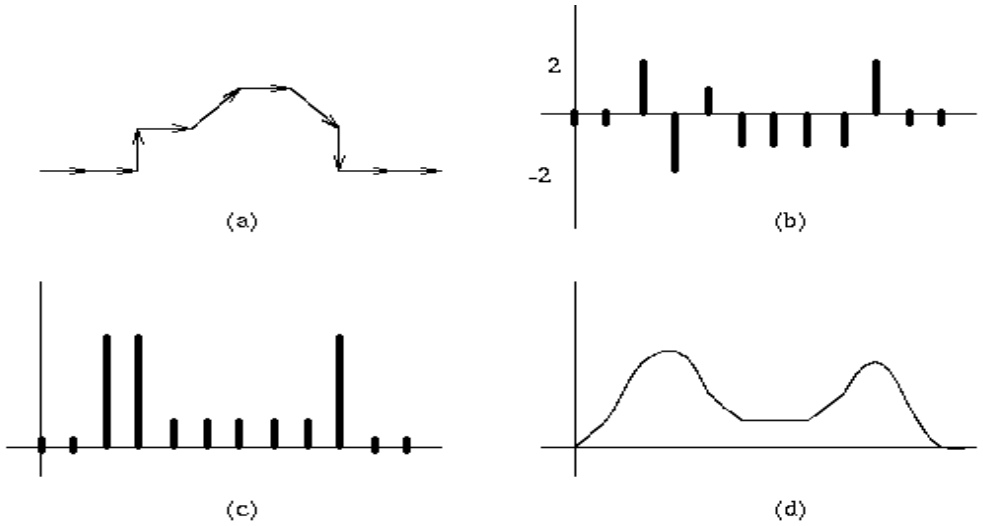


Figure 6.8 *Bending energy: (a) Chain code 0, 0, 2, 0, 1, 0, 7, 6, 0, 0, (b) curvature 0, 2, -2, 1, -1, -1, -1, 2, 0, (c) sum of squares gives the bending energy, (d) smoothed version.*

- ***Signature***

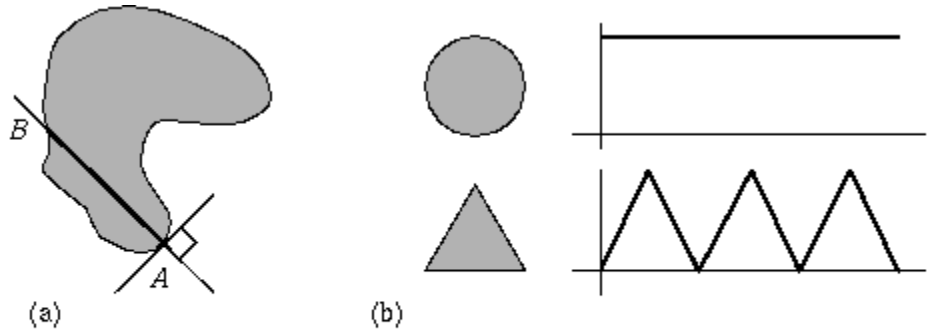


Figure 6.9 *Signature: (a) Construction, (b) signatures for a circle and a triangle.*

- ***Chord***
  - Chord is a line joining any two points of the region boundary is a chord.
  - Let b(x,y)=1 represent the contour points, and b(x,y)=0 represent all other points.

$$h(\Delta x, \Delta y) = \int\int b(x,y)b(x+\Delta x, y+\Delta y)dxdy \qquad (6.2)$$

$$h(\Delta x, \Delta y) = \sum_i \sum_j b(i,j)b(i+\Delta x, j+\Delta y) \qquad (6.3)$$

- Rotation-independent radial distribution:

$$h_r(r) = \int_{-\pi/2}^{\pi/2} h(\Delta x, \Delta y) r d\theta \qquad (6.4)$$

- The angular distribution h_a(theta) is independent of scale, while rotation causes a proportional offset

$$h_a(\theta) = \int_0^{\max(r)} h(\Delta x, \Delta y) dr \qquad (6.5)$$

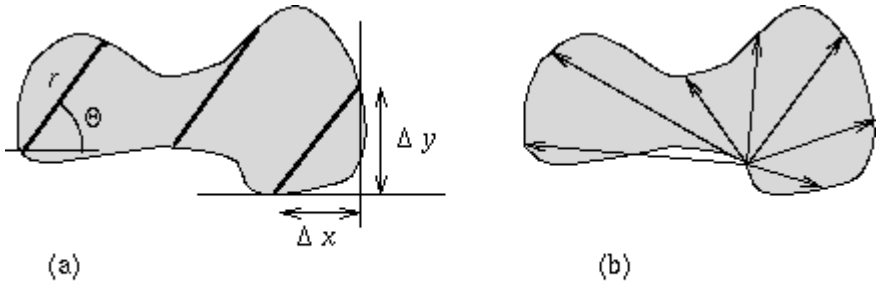


Figure 6.10 *Chord distribution.*

REGION-BASED SHAPE REPRESENTATION AND DESCRIPTION:

- A large group of shape description techniques is represented by heuristic approaches which yield acceptable results in description of simple shapes.
- Heuristic region descriptors:
  - ✓ area,
  - ✓ rectangularity,
  - ✓ elongatedness,
  - ✓ direction,
  - ✓ compactness,
- These descriptors cannot be used for region reconstruction and do not work for more complex shapes.
- Procedures based on region decomposition into smaller and simpler subregions must be applied to describe more complicated regions, then subregions can be described separately using heuristic approaches.

A. SIMPLE SCALAR REGION DESCRIPTORS:

- Area is given by the number of pixels of which the region consists.
- The *real* area of each pixel may be taken into consideration to get the *real* size of a region.
- If an image is represented as a rectangular raster, simple counting of region pixels will provide its area.
- If the image is represented by a quadtree, then:

Algorithm 6.4: Calculating area in quadtrees

1. Set all region area variables to zero, and determine the global quadtree depth H; for example, the global quadtree depth is $H = 8$ for a 256×256 image.

2. Search the tree in a systematic way. If a leaf node at a depth h has a non-zero label, proceed to step (3).

3. Compute:

$$area[region\_label] = area[region\_label] + 4^{(H-h)}$$

4. The region areas are stored in variables $area[region\_label]$.

- The region can also be represented by n polygon vertices

$$area = \frac{1}{2}\left|\sum_{k=0}^{n-1}(i_k j_{k+1} - i_{k+1} j_k)\right| \qquad (6.33)$$

the sign of the sum represents the polygon orientation.

- If the region is represented by the (anti-clockwise) Freeman chain code the following algorithm provides the area

Algorithm 6.5: Region area calculation from Freeman 4-connectivity chain code representation

1. Set the region *area* to zero. Assign the value of the starting point *i* co-ordinate to the variable *vertical_position*.

2. For each element of the chain code (values 0, 1, 2, 3) do

```
switch(code) {
    case 0:
        area := area - vertical_position;
        break;
    case 1:
        vertical_position := vertical_position + 1;
        break;
    case 2:
        area := area + vertical_position;
        break;
    case 3:
        vertical_position := vertical_position - 1;
        break;
}
```

3. If all boundary chain elements have been processed, the region area is stored in the variable *area*.

Euler's number:

- (Sometimes called **Genus** or the **Euler-Poincare characteristic**) describes a simple topologically invariant property of the object.

  - ✓ S is the number of contiguous parts of an object and N is the number of holes in the object (an object can consist of more than one region).

$$\vartheta = S - N \qquad (6.34)$$

Projections:

- Horizontal and vertical region projections

$$p_h(i) = \sum_j f(i,j), \qquad p_v(j) = \sum_i f(i,j). \qquad (6.35)$$

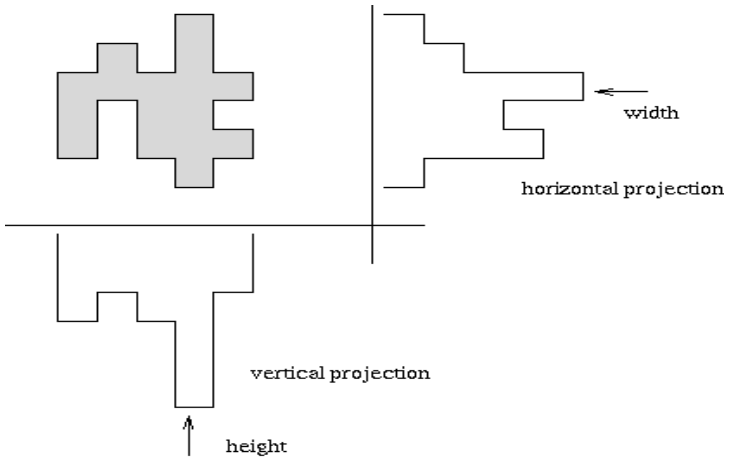


Figure 6.22 *Projections.*

Eccentricity:

- The simplest is the ratio of major and minor axes of an object.

Elongatedness:

- A ratio between the length and width of the region bounding rectangle.

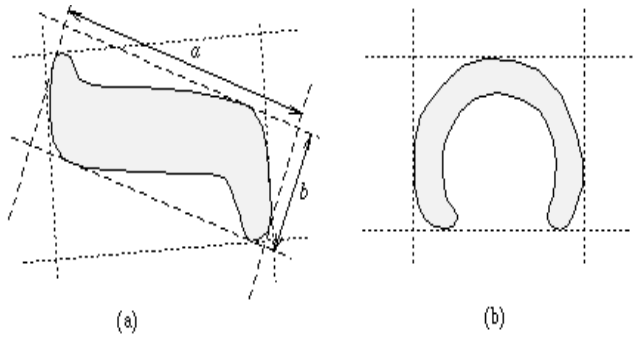


Figure 6.24 *Elongatedness: (a) Bounding rectangle gives acceptable results, (b) bounding rectangle cannot represent elongatedness.*

- This criterion cannot succeed in curved regions, for which the evaluation of elongatedness must be based on maximum region thickness.
- Elongatedness can be evaluated as a ratio of the region area and the square of its thickness.
- The maximum region thickness (holes must be filled if present) can be determined as the number d of erosion steps that may be applied before the region totally disappears.

$$elongatedness = \frac{area}{(2d)^2} \qquad (6.36)$$

Rectangularity:

- Let F_k be the ratio of region area and the area of a bounding rectangle, the rectangle having the direction k. The rectangle direction is turned in discrete steps as before, and rectangularity measured as a maximum of this ratio F_k

$$rectangularity = \max_k(F_k) \qquad (6.38)$$

Direction:

- Direction is a property which makes sense in elongated regions only.
- If the region is elongated, direction is the direction of the longer side of a minimum bounding rectangle.
- If the shape moments are known, the direction \theta can be computed as

$$\theta = \frac{1}{2}tan^{-1}(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}) \qquad (6.39)$$

- Elongatedness and rectangularity are independent of linear transformations -- translation, rotation, and scaling.
- Direction is independent on all linear transformations which do not include rotation.
- Mutual direction of two rotating objects is rotation invariant.

Compactness:

- Compactness is independent of linear transformations

$$compactness = \frac{(region\_border\_length)^2}{area} \qquad (6.40)$$

- The most compact region in a Euclidean space is a circle.
- Compactness assumes values in the interval [1,infty) in digital images if the boundary is defined as an inner boundary, while using the outer boundary, compactness assumes values in the interval [16,infty).
- Independence from linear transformations is gained only if an outer boundary representation is used.

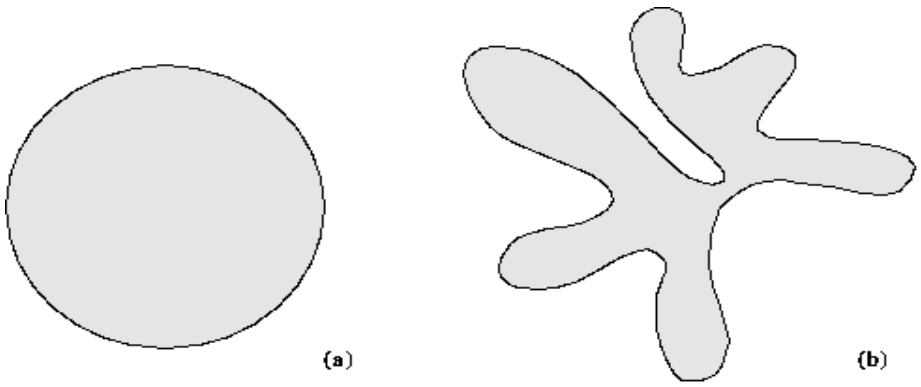


Figure 6.25 *Compactness: (a) Compact, (b) non-compact.*

Moments:

- Region moment representations interpret a normalized gray level image function as a probability density of a 2D random variable.
- Properties of this random variable can be described using statistical characteristics - **moments**.
- Assuming that non-zero pixel values represent regions, moments can be used for binary or gray level region description.
- A moment of order (p+q) is dependent on scaling, translation, rotation, and even on gray level transformations and is given by

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x,y)\, dx\, dy \qquad (6.41)$$

- In digitized images we evaluate sums

$$m_{pq} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} i^p j^q f(i,j) \qquad (6.42)$$

- where x,y,i,j are the region point co-ordinates (pixel co-ordinates in digitized images).

- Translation invariance can be achieved if we use the central moments

$$\mu_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - x_c)^p (y - y_c)^q f(x, y)\, dx\, dy \qquad (6.43)$$

- or in digitized images

$$\mu_{pq} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} (i - x_c)^p (j - y_c)^q f(i, j) \qquad (6.44)$$

- where x_c, y_c are the co-ordinates of the region's centroid

$$x_c = \frac{m_{10}}{m_{00}} \qquad (6.45)$$

$$y_c = \frac{m_{01}}{m_{00}}$$

- In the binary case, m_00 represents the region area.
  - ✓ Scale invariant features can also be found in scaled central moments

$$\eta_{pq} = \frac{\mu'_{pq}}{(\mu'_{00})^\gamma} \qquad (6.46)$$

$$\gamma = \frac{p + q}{2} + 1$$

$$\mu'_{pq} = \frac{\mu_{pq}}{\alpha^{(p+q+2)}}$$

and normalized unscaled central moments

$$\vartheta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^\gamma} \qquad (6.47)$$

- Rotation invariance can be achieved if the co-ordinate system is chosen such that mu_11 = 0.
- A less general form of invariance is given by seven rotation, translation, and scale invariant moment characteristics

$$\varphi_1 = \vartheta_{20} + \vartheta_{02} \qquad (6.48)$$

$$\varphi_2 = (\vartheta_{20} - \vartheta_{02})^2 + 4\vartheta_{11}^2 \tag{6.49}$$

$$\varphi_3 = (\vartheta_{30} - 3\vartheta_{12})^2 + (3\vartheta_{21} - \vartheta_{03})^2 \tag{6.50}$$

$$\varphi_4 = (\vartheta_{30} + \vartheta_{12})^2 + (\vartheta_{21} + \vartheta_{03})^2 \tag{6.51}$$

$$\begin{aligned}\varphi_5 = \quad & (\vartheta_{30} - 3\vartheta_{12})(\vartheta_{30} + \vartheta_{12})[(\vartheta_{30} + \vartheta_{12})^2 - 3(\vartheta_{21} + \vartheta_{03})^2] + \\ & (3\vartheta_{21} - \vartheta_{03})(\vartheta_{21} + \vartheta_{03})[3(\vartheta_{30} + \vartheta_{12})^2 - (\vartheta_{21} + \vartheta_{03})^2]\end{aligned} \tag{6.52}$$

$$\varphi_6 = (\vartheta_{20} - \vartheta_{02})[(\vartheta_{30} + \vartheta_{12})^2 - (\vartheta_{21} + \vartheta_{03})^2] + 4\vartheta_{11}(\vartheta_{30} + \vartheta_{12})(\vartheta_{21} + \vartheta_{03}) \tag{6.53}$$

$$\begin{aligned}\varphi_7 = \quad & (3\vartheta_{21} - \vartheta_{03})(\vartheta_{30} + \vartheta_{12})[(\vartheta_{30} + \vartheta_{12})^2 - 3(\vartheta_{21} + \vartheta_{03})^2] - \\ & (\vartheta_{30} - 3\vartheta_{12})(\vartheta_{21} + \vartheta_{03})[3(\vartheta_{30} + \vartheta_{12})^2 - (\vartheta_{21} + \vartheta_{03})^2]\end{aligned} \tag{6.54}$$

- While the seven moment characteristics presented above were shown to be useful, they are only invariant to translation, rotation, and scaling.
- A complete set of four affine moment invariants derived from second- and third-order moments is

$$I_1 = \frac{\mu_{20}\mu_{02} - \mu_{11}^2}{\mu_{00}^4} \tag{6.55}$$

$$I_2 = \frac{\mu_{30}^2\mu_{03}^2 - 6\mu_{30}\mu_{21}\mu_{12}\mu_{03} + 4\mu_{30}\mu_{12}^3 + 4\mu_{21}^3\mu_{03} - 3\mu_{21}^2\mu_{12}^2}{\mu_{00}^{10}} \tag{6.56}$$

$$I_3 = \frac{\mu_{20}(\mu_{21}\mu_{03} - \mu_{12}^2) - \mu_{11}(\mu_{30}\mu_{03} - \mu_{21}\mu_{12}) + \mu_{02}(\mu_{30}\mu_{12} - \mu_{21}^2)}{\mu_{00}^7} \tag{6.57}$$

$$\begin{aligned}I_4 = \quad & (\mu_{20}^3\mu_{03}^2 - 6\mu_{20}^2\mu_{11}\mu_{12}\mu_{03} - 6\mu_{20}^2\mu_{02}\mu_{21}\mu_{03} + 9\mu_{20}^2\mu_{02}\mu_{12}^2 \\ & + 12\mu_{20}\mu_{11}^2\mu_{21}\mu_{03} + 6\mu_{20}\mu_{11}\mu_{02}\mu_{30}\mu_{03} - 18\mu_{20}\mu_{11}\mu_{02}\mu_{21}\mu_{12} \\ & - 8\mu_{11}^3\mu_{30}\mu_{03} - 6\mu_{20}\mu_{02}^2\mu_{30}\mu_{12} + 9\mu_{20}\mu_{02}^2\mu_{21}^2 \\ & + 12\mu_{11}^2\mu_{02}\mu_{30}\mu_{12} - 6\mu_{11}\mu_{02}^2\mu_{30}\mu_{21} + \mu_{02}^3\mu_{30}^2)/\mu_{00}^{11}\end{aligned} \tag{6.58}$$

- All moment characteristics are dependent on the linear gray level transformations of regions; to describe region shape properties, we work with binary image data (f(i,j)=1 in region pixels) and dependence on the linear gray level transform disappears.
- Moment characteristics can be used in shape description even if the region is represented by its boundary.

- A closed boundary is characterized by an ordered sequence z(i) that represents the Euclidean distance between the centroid and all N boundary pixels of the digitized shape.
- No extra processing is required for shapes having spiral or concave contours.
- Translation, rotation, and scale invariant one-dimensional normalized contour sequence moments can be estimated as

$$m_r = \frac{1}{N}\sum_{i=1}^{N}[z(i)]^r \tag{6.59}$$

$$\mu_r = \frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^r \tag{6.60}$$

- The r-th normalized contour sequence moment and normalized central contour sequence moment are defined as

$$\overline{m}_r = \frac{m_r}{(\mu_2)^{r/2}} = \frac{\frac{1}{N}\sum_{i=1}^{N}[z(i)]^r}{[\frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^2]^{r/2}} \tag{6.61}$$

$$\overline{\mu}_r = \frac{\mu_r}{(\mu_2)^{r/2}} = \frac{\frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^r}{[\frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^2]^{r/2}} \tag{6.62}$$

- Less noise-sensitive results can be obtained from the following shape descriptors

$$F_1 = \frac{(\mu_2)^{1/2}}{m_1} = \frac{[\frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^2]^{1/2}}{\frac{1}{N}\sum_{i=1}^{N} z(i)} \tag{6.63}$$

$$F_2 = \frac{\mu_3}{(\mu_2)^{3/2}} = \frac{\frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^3}{[\frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^2]^{3/2}} \tag{6.64}$$

$$F_3 = \frac{\mu_4}{(\mu_2)^2} = \frac{\frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^4}{[\frac{1}{N}\sum_{i=1}^{N}[z(i) - m_1]^2]^2} \tag{6.65}$$

$$F_4 = \overline{\mu}_5 \tag{6.66}$$

CONVEX HULL:

- A region R is convex if and only if for any two points x_1, x_2 from R, the whole line segment defined by its end-points x_1, x_2 is inside the region R.
- The convex hull of a region is the smallest convex region H which satisfies the condition R is a subset of H.

- The convex hull has some special properties in digital data which do not exist in the continuous case. For instance, concave parts can appear and disappear in digital data due to rotation, and therefore the convex hull is not rotation invariant in digital space.
- The convex hull can be used to describe region shape properties and can be used to build a tree structure of region concavity.

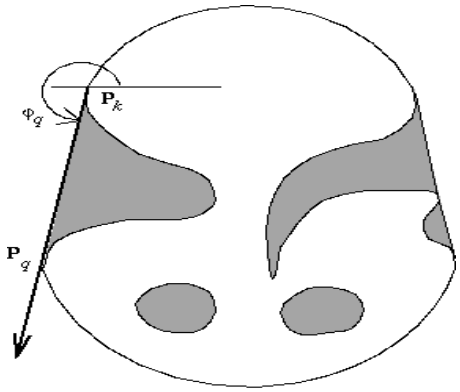


Figure 6.26 *Convex hull.*

- A discrete convex hull can be defined by the following algorithm which may also be used for convex hull construction.
    - This algorithm has complexity O(n^2) and is presented here as an intuitive way of detecting the convex hull.

Algorithm 6.6: Region convex hull construction

1. Find all pixels of a region R with the minimum row co-ordinate; among them, find the pixel P_1 with the minimum column co-ordinate.
   Assign $\mathbf{P}_k = \mathbf{P}_1$, $\mathbf{v} = (0, -1)$; the vector $\mathbf{v}$ represents the direction of the previous line segment of the convex hull.

2. Search the region boundary in an anti-clockwise direction (Algorithm **??**) and compute the angle orientation φ_n for every boundary point $\mathbf{P}_n$ which lies after the point $\mathbf{P}_1$ (in the direction of boundary search – see Figure 6.26). The angle orientation φ_n is the angle of vector $\mathbf{P}_k\mathbf{P}_n$. The point $\mathbf{P}_q$ satisfying the condition $\varphi_q = \min_n \varphi_n$ is an element (vertex) of the region convex hull.

3. Assign $\mathbf{v} = \mathbf{P}_k\mathbf{P}_q$, $\mathbf{P}_k = \mathbf{P}_q$.

4. Repeat steps (2) and (3) until $\mathbf{P}_k = \mathbf{P}_1$.

- More efficient algorithms exist, especially if the object is defined by an ordered sequence of n vertices representing a polygonal boundary of the object.
- If the polygon P is a *simple* polygon (self-non-intersecting polygon) which is always the case in a polygonal representation of object borders, the convex hull may be found in linear time O(n).
- In the past two decades, many linear-time convex hull detection algorithms have been published, however more than half of them were later discovered to be incorrect with counter-examples published.
- The simplest correct convex hull algorithm was developed by Melkman and is now discussed further.
  - Let the polygon for which the convex hull is to be determined be a simple polygon P = v_1, v_2, ... v_n and let the vertices be processed in this order.
  - For any three vertices x,y,z in an ordered sequence, a directional function delta may be evaluated

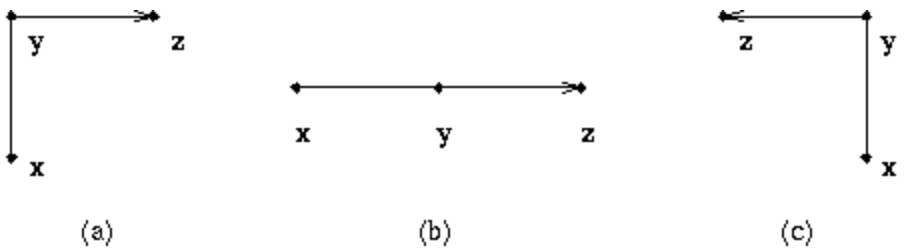


Figure 6.27 *Directional function δ: (a) $\delta(\mathbf{x},\mathbf{y},\mathbf{z}) = 1$, (b) $\delta(\mathbf{x},\mathbf{y},\mathbf{z}) = 0$, (c) $\delta(\mathbf{x},\mathbf{y},\mathbf{z}) = -1$.*

- The main data structure H is a list of vertices (deque) of polygonal vertices already processed.
- The current contents of H represent the convex hull of the currently processed part of the polygon, and after the detection is completed, the convex hull is stored in this data structure.
- Therefore, H always represents a closed polygonal curve, H={d_b, ... ,d_t} where d_b points to the bottom of the list and d_t points to its top.
- Note that d_b and d_t always refer to the same vertex simultaneously representing the first and the last vertex of the closed polygon.
- Main ideas of the algorithm:
  - The first three vertices A,B,C from the sequence P form a triangle (if not collinear) and this triangle represents a convex hull of the first three vertices.
  - The next vertex D in the sequence is then tested for being located inside or outside the current convex hull.
  - If D is located inside, the current convex hull does not change.
  - If D is outside of the current convex hull, it must become a new convex hull vertex and based on the current convex hull shape, none, one, or several vertices must be removed from the current convex hull.
  - This process is repeated for all remaining vertices in the sequence P.

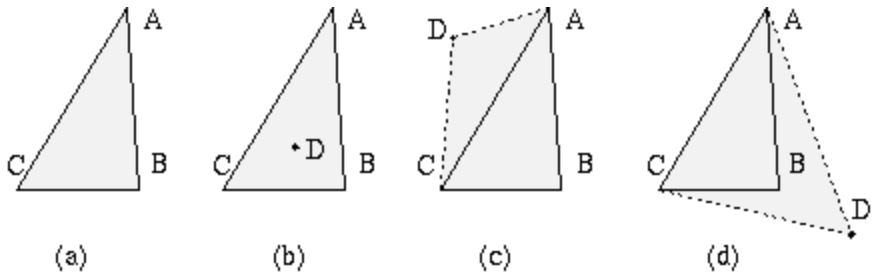


Figure 6.28 *Convex hull detection: (a) First three vertices A, B, C form a triangle, (b) if the next vertex D is positioned inside the current convex hull ABC, current convex hull does not change, (c) if the next vertex D is outside of the current convex hull, it becomes a new vertex of the new current convex hull $ABCDA$, (d) in this case, vertex B must be removed from the current convex hull and the new current convex hull is $ADCA$.*

- The variable v refers to the input vertex under consideration, and the following operations are defined:

push v	:	$t := t + 1; \quad d_t \rightarrow \mathbf{v}$
pop d_t	:	$t := t - 1$
insert v	:	$b := b - 1; \quad d_b \rightarrow \mathbf{v}$
remove d_b	:	$b := b + 1$
input v	:	next vertex is entered from sequence P, if P is empty, stop.

- The algorithm is then;

Algorithm 6.7: Simple polygon convex hull detection

1. Initialize;
   . $t := -1$;
   . $b := 0$;
   . input $\mathbf{v}_1$; input $\mathbf{v}_2$; input $\mathbf{v}_3$;
   . if ($\delta(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3) > 0$)
   . { push $\mathbf{v}_1$; push $\mathbf{v}_2$; }
   . else
   . { push $\mathbf{v}_2$; push $\mathbf{v}_1$; }
   . push $\mathbf{v}_3$;
   . insert $\mathbf{v}_3$;

2. If the next vertex $\mathbf{v}$ is inside the current convex hull H, enter and check a new vertex; otherwise process steps (3) and (4);
   . input $\mathbf{v}$;
   . while ($\delta(\mathbf{v}, d_b, d_{b+1}) \geq 0$ AND $\delta(d_{i-1}, d_i, \mathbf{v}) \geq 0$)
   . input $\mathbf{v}$;

3. Rearrange vertices in H, top of the list.
   . while ($\delta(d_{i-1}, d_i, \mathbf{v}) \leq 0$)
   . pop d_i;
   . push $\mathbf{v}$;

4. Rearrange vertices in H, bottom of the list.
   . while ($\delta(\mathbf{v}, d_b, d_{b+1}) \leq 0$)
   . remove d_b;
   . insert $\mathbf{v}$;
   . go to step (2);

- The algorithm as presented may be difficult to follow, however, a less formal version would be impossible to implement.
- The following example makes the algorithm more understandable.

In the second step, vertex D is entered (Figure 6.29b);

$$\begin{array}{rcl} \delta(D, d_b, d_{b+1}) & = & \delta(D, C, A) = 1 > 0 \\ \delta(d_{i-1}, d_i, D) & = & \delta(B, C, D) = -1 < 0 \end{array}$$

Based on the values of the directional function δ, in this case, no other vertex is entered during this step. Step (3) results in the following current convex hull H;

$$\delta(B, C, D) = -1 \longrightarrow \texttt{pop } d_i \longrightarrow$$

$t, b \ldots$		-1	0	1	2
H	$=$	C	A	B	C
		d_b		d_i	

$$\delta(A, B, D) = -1 \longrightarrow \texttt{pop } d_i \longrightarrow$$

$t, b \ldots$		-1	0	1	2
H	$=$	C	A	B	C
		d_b	d_i		

$$\delta(C, A, D) = 1 \longrightarrow \texttt{push } D \longrightarrow$$

$t, b \ldots$		-1	0	1	2
H	$=$	C	A	D	C
		d_b		d_i	

In step (4) – Figure 6.29c;

$$\delta(D, C, A) = 1 \longrightarrow \texttt{insert } D \longrightarrow$$

$t, b \ldots$		-2	-1	0	1	2
H	$=$	D	C	A	D	C
		d_b			d_i	

Go to step (2); vertex E is entered – Figure 6.29d;

$$\begin{array}{rcl} \delta(E, D, C) & = & 1 > 0 \\ \delta(A, D, E) & = & 1 > 0 \end{array}$$

- A new vertex should be entered from P, however there is no unprocessed vertex in the sequence P and the convex hull generating process stops.
- The resulting convex hull is defined by the sequence H={d_b, ... ,d_t}={D,C,A,D} which represents a polygon DCAD, always in the clockwise direction.

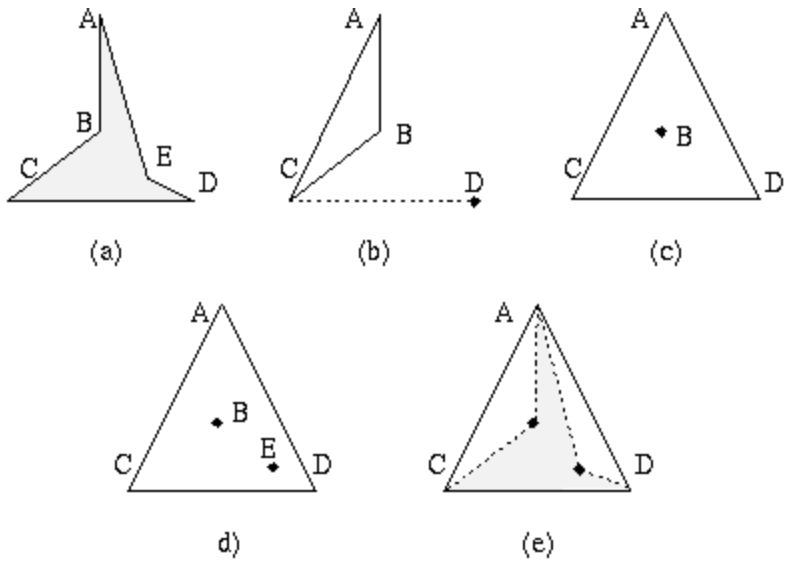


Figure 6.29 *Example of convex hull detection: (a) The processed region – polygon ABCDEA, (b) vertex D is entered and processed, (c) vertex D becomes a new vertex of the current convex hull ADC, (d) vertex E is entered and processed, E does not become a new vertex of the current convex hull, (e) the resulting convex hull DCAD.*

- A **region concavity tree** is generated recursively during the construction of a convex hull.
- A convex hull of the whole region is constructed first, and convex hulls of concave residua are found next.
- The resulting convex hulls of concave residua of the regions from previous steps are searched until no concave residuum exists.
- The resulting tree is a shape representation of the region.

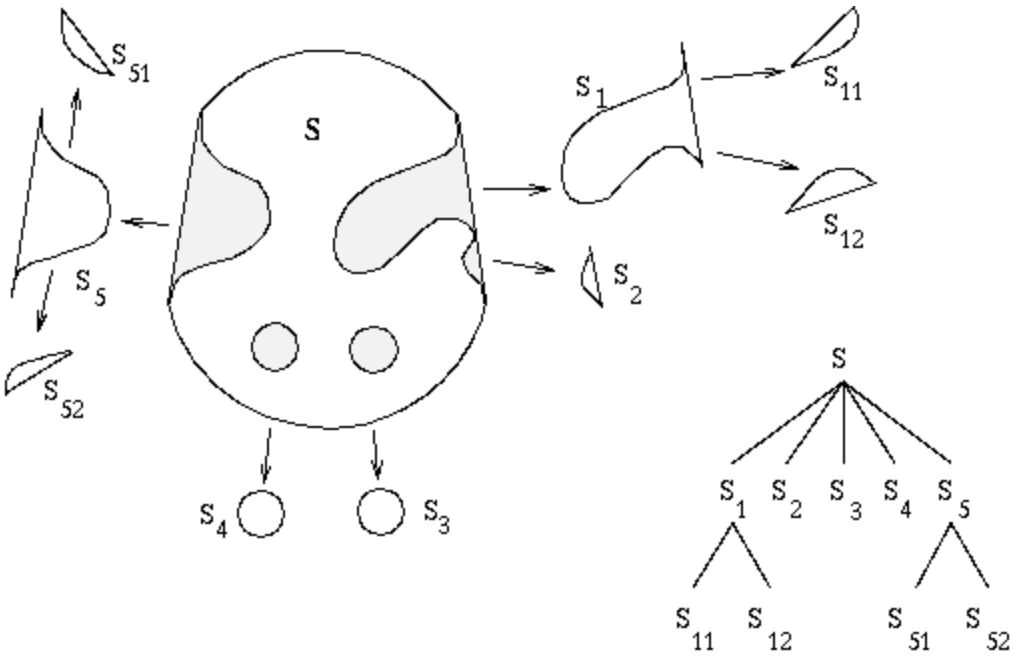


Figure 6.30 *Concavity tree construction: (a) Convex hull and concave residua, (b) concavity tree.*

GRAPH REPRESENTATION BASED ON REGION SKELETON:

- Objects are represented by a planar graph with nodes representing subregions resulting from region decomposition, and region shape is then described by the graph properties.
- There are two general approaches to acquiring a graph of subregions:
  - ✓ The first one is region thinning leading to the **region skeleton**, which can be described by a graph.
  - ✓ The second option starts with the **region decomposition** into subregions, which are then represented by nodes while arcs represent neighborhood relations of subregions.
- Graphical representation of regions has many advantages; the resulting graphs
  - ✓ are translation and rotation invariant; position and rotation can be included in the graph definition
  - ✓ are insensitive to small changes in shape
  - ✓ are highly invariant with respect to region magnitude
  - ✓ generate a representation which is understandable
  - ✓ can easily be used to obtain the information-bearing features of the graph

Algorithm 6.8: Skeleton by thinning

1. Let R be the set of region pixels, $H_i(R)$ its inner boundary, and $H_o(R)$ its outer boundary. Let $S(R)$ be a set of pixels from the region R which have all their neighbors in 8-connectivity either from the inner boundary $H_i(R)$ or from the background – from the residuum of R. Assign $R_{old} = R$.

2. Construct a region R_{new} which is a result of one-step thinning as follows

$$R_{new} = S(R_{old}) \cup [R_{old} - H_i(R_{old})] \cup [H_o(S(R_{old})) \cap R_{old}]$$

3. If $R_{new} = R_{old}$, terminate the iteration and proceed to step (4). Otherwise assign $R_{old} = R_{new}$ and repeat step (2).

4. R_{new} is a set of skeleton pixels, the skeleton of the region R.

- Steps of this algorithm are illustrated in the next Figure.
- If there are skeleton segments which have a thickness of two, one extra step can be added to reduce those to a thickness of one, although care must be taken not to break the skeleton connectivity.

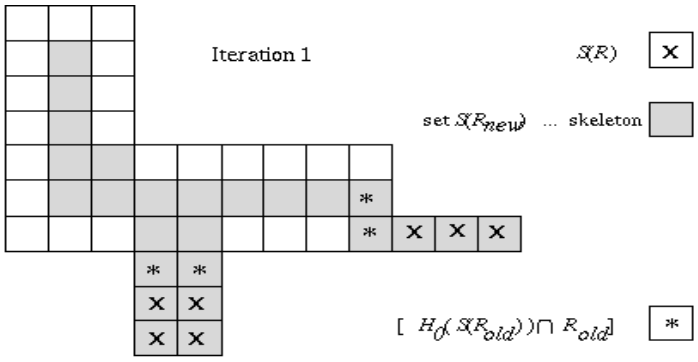


Figure 6.31 *Skeleton by thinning (Algorithm 6.8).*

- Thinning is generally a time-consuming process, although sometimes it is not necessary to look for a skeleton, and one side of a parallel boundary can be used for skeleton-like region representation. Mathematical morphology is a powerful tool used to find the region skeleton.
- Thinning procedures often use a medial axis transform to construct a region skeleton. Under the medial axis definition, the skeleton is the set of all region points which have the same minimum distance from the region boundary for at least two separate boundary points.

- Such a skeleton can be constructed using a distance transform which assigns a value to each region pixel representing its (minimum) distance from the region's boundary. The skeleton can be determined as a set of pixels whose distance from the region's border is locally maximal.
- Every skeleton element can be accompanied by information about its distance from the boundary -- this gives the potential to reconstruct a region as an envelope curve of circles with center points at skeleton elements and radii corresponding to the stored distance values. Small changes in the boundary may cause serious changes in the skeleton.
- This sensitivity can be removed by first representing the region as a polygon, then constructing the skeleton. Boundary noise removal can be absorbed into the polygon construction.
- A multi-resolution approach to skeleton construction may also result in decreased sensitivity to boundary noise. Similarly, the approach using the Marr-Hildreth edge detector with varying smoothing parameter facilitates scale-based representation of the region's skeleton.

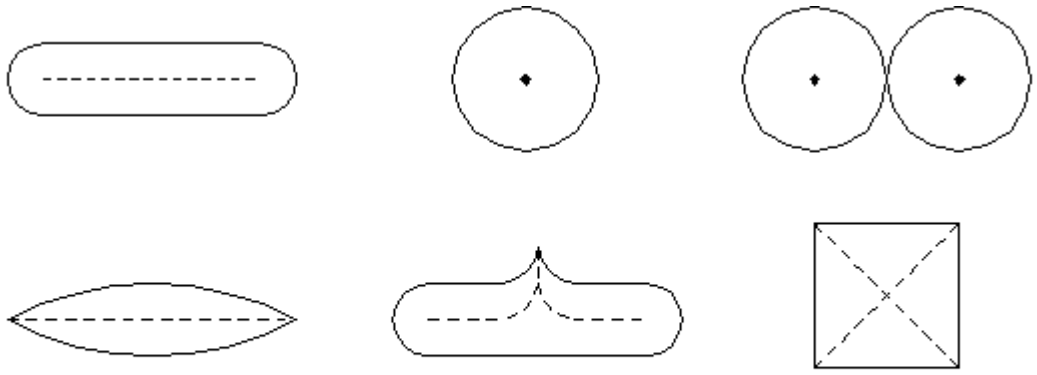


Figure 6.32 *Region skeletons; small changes in border can have a significant effect on the skeleton.*

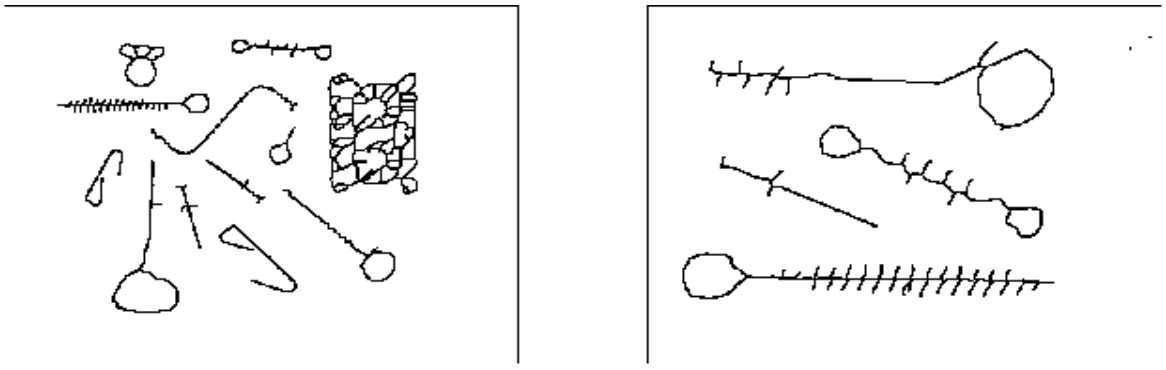


Figure 6.33 *Region skeletons, see Figures ??a and 6.2a for original images; thickened for visibility.*

- Skeleton construction algorithms do not result in graphs but the transformation from skeletons to graphs is relatively straightforward.
- Consider first the medial axis skeleton, and assume that a minimum radius circle has been drawn from each point of the skeleton which has at least one point common with a region boundary.
- Let **contact** be each contiguous subset of the circle which is common to the circle and to the boundary.
- If a circle drawn from its center A has one contact only, A is a skeleton end-point.
- If the point A has two contacts, it is a normal skeleton point.
- If A has three or more contacts, the point A is a skeleton node-point.

Algorithm 6.9: Region graph construction from skeleton

1. Assign a point description to all skeleton points – end-point, node-point, normal-point.
2. Let graph node-points be all end-points and node-points. Connect any two graph nodes by a graph edge if they are connected by a sequence of normal-points in the region skeleton.

- It can be seen that boundary points of high curvature have the main influence on the graph.
- They are represented by graph nodes, and therefore influence the graph structure.
- If other than medial axis skeletons are used for graph construction, end-points can be defined as skeleton points having just one skeleton neighbor, normal-points as having two skeleton neighbors, and node-points as having at least three skeleton neighbors.
- It is no longer true that node-points are never neighbors and additional conditions must be used to decide when node-points should be represented as nodes in a graph and when they should not.

REGION DECOMPOSITION:

- The decomposition approach is based on the idea that shape recognition is a hierarchical process.
- Shape primitives are defined at the lower level, primitives being the simplest elements which form the region.
- A graph is constructed at the higher level - nodes result from primitives, arcs describe the mutual primitive relations.
- Convex sets of pixels are one example of simple shape primitives.

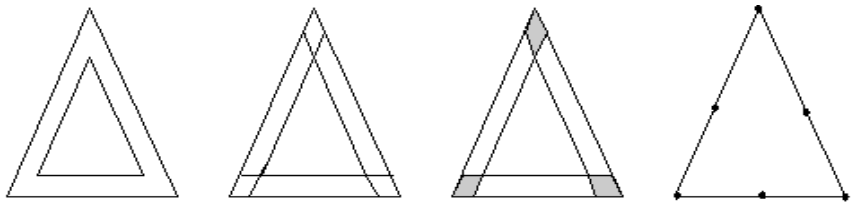


Figure 6.34 *Region decomposition: (a) Region, (b) primary regions, (c) primary subregions and kernels, (d) decomposition graph.*

- The solution to the decomposition problem consists of two main steps:
  - ✓ The first step is to segment a region into simpler subregions (primitives) and the second is the analysis of primitives.
  - ✓ Primitives are simple enough to be successfully described using simple scalar shape properties.
  - ✓ If subregions are represented by polygons, graph nodes bear the following information;
    - ❖ Node type representing primary subregion or kernel.
    - ❖ Number of vertices of the subregion represented by the node.
    - ❖ Area of the subregion represented by the node.
    - ❖ Main axis direction of the subregion represented by the node.
    - ❖ Center of gravity of the subregion represented by the node.
  - ✓ If a graph is derived using attributes 1-4, the final description is translation invariant.
  - ✓ A graph derived from attributes 1-3 is translation and rotation invariant.
  - ✓ Derivation using the first two attributes results in a description which is size invariant in addition to possessing translation and rotation invariance.

REGION NEIGHBORHOOD GRAPHS:

- Any time region decomposition into subregions or an image decomposition into regions is available, the region or image can be represented by a region neighborhood graph (the region adjacency graph being a special case).
- This graph represents every region as a graph node, and nodes of neighboring regions are connected by edges.
- A region neighborhood graph can be constructed from a quadtree image representation, from run-length encoded image data, etc.

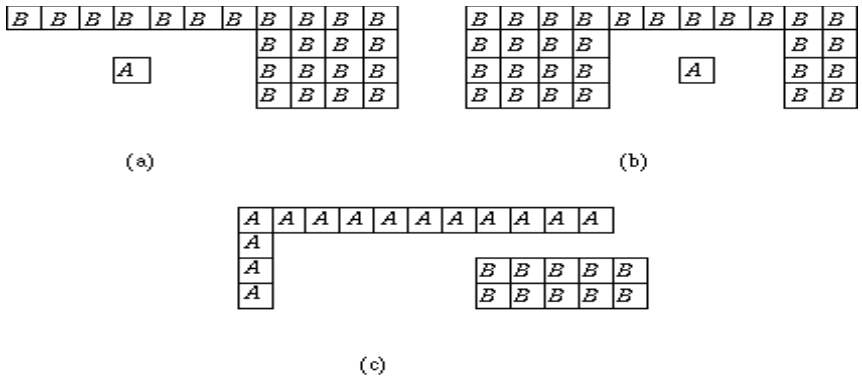


Figure 6.35 *Binary relation* to be left of*; see text*.

- Very often, the relative position of two regions can be used in the description process -- for example, a region A may be positioned to the **left of** a region B, or **above** B, or **close to** B, or a region C may lie **between** regions A and B, etc.
- We know the meaning of all of the given relations if A,B,C are points, but, with the exception of the relation **to be close**, they can become ambiguous if A,B,C are regions.
- For instance, human observers are generally satisfied with the definition:
  - ✓ The center of gravity of A must be positioned to the left of the leftmost point of B and (logical AND) the rightmost pixel of A must be left of the rightmost pixel of B.

QUESTION BANK

5 MARKS:

1. Explain the segmentation in detail.
2. Explain the thresholding modifications in detail.
3. Explain the multi-spectral thresholding in detail.
4. Explain the hierarchical thresholding in detail.
5. Explain the edge relaxation in detail.
6. Explain the border tracing in detail **(April 2012)**.
7. Explain the region merging in detail.
8. Explain the water shade segmentation in detail.
9. Explain the matching in detail **(April 2012)**.
10. Explain the region identification in detail.
11. Explain the chain codes in detail.
12. Explain the simple scalar region descriptors in detail.
13. Explain the region decomposition in detail.

10 MARKS:

1. Explain the thresholding in detail.
2. Explain the threshold detection methods in detail **(April 2012)**.
3. Explain the edge-based segmentation in detail.
4. Explain the region-based segmentation in detail.
5. Explain the splitting and merging in detail.
6. Explain the shape representation and description in detail.
7. Explain the contour-based shape representation and description in detail **(April 2012)**.
8. Explain the region-based shape representation and description in detail.
9. Explain the graph representation based on region skeleton in detail.
10. Explain the simple geometric border representation in detail.
11. Explain the region neighborhood graphs in detail.
12. Explain the convex hull in detail.

UNIT-IV

STATISTICAL PATTERN RECOGNITION:

Knowledge Representation:

Knowledge Representation (KR) has long been considered one of the principal elements of Artificial Intelligence, and a critical part of all problem solving. The subfields of KR range from the purely philosophical aspects of epistemology to the more practical problems of handling huge amounts of data.

This diversity is unified by the central problem of encoding human knowledge - in all its various forms - in such a way that the knowledge can be used. This goal is perhaps best summarized in the *Knowledge Representation Hypothesis:*

Any mechanically embodied intelligent process will be comprised of structural ingredients that a) we as external observers naturally take to represent a propositional account of the knowledge that the overall process exhibits, and b) independent of such external semantical attribution, play a formal but causal and essential role in engendering the behavior that manifests that knowledge.

The syntax of representation specifies the symbols that may be used and the ways that thaey may be arranged.

The semantics of representation specifies how meaning is embodied in the symbols and the symbols arranged allowed to the syntax.

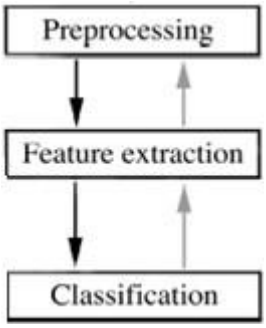
A representation is a set of syntax and semantic conventions that make it possible to describe things.

- The study of how machines can observe the environment,
  - learn to distinguish *patterns* of interest from their background, and
  - Make sound and reasonable decisions about the *categories* of the patterns.
- Pattern is an entity, vaguely defined, that could be given a name.
- For example, a pattern could be
  - A fingerprint images
  - A handwritten cursive word
  - A human face
  - A speech signal
- The design of a pattern recognition system essentially involves the following three aspects:
  - data acquisition
  - data representation
  - decision making

Classification:

Once a feature selection or a classification procedure finds a proper representation, a classifier can be designed using a number of possible approaches.

In practice, the choice of a classifier is a difficult problem and it is often based on which classifier(s) happen to be available, best known, to the user.



Linear classifiers:

A large number of algorithms for classification can be phrased in terms of a linear function that assigns a score to each possible category *k* by combining the feature vector of an instance with a vector of weights, using a dot product. The predicted category is the one with the highest score. This type of score function is known as a linear predictor function and has the following general form:

$$\text{score}(\mathbf{X}_i, k) = \boldsymbol{\beta}_k \cdot \mathbf{X}_i,$$

Where $\mathbf{X}_i$ is the feature vector for instance i, $\boldsymbol{\beta}_k$ is the vector of weights corresponding to category k, and score ($\mathbf{X}_i$, k) is the score associated with assigning instance i to category k. In discrete choice theory, where instances represent people and categories represent choices, the score is considered the utility associated with person i choosing category k.

Algorithms with this basic setup are known as linear classifiers. What distinguishes them is the procedure for determining (training) the optimal weights/coefficients and the way that the score is interpreted.

Examples of such algorithms:

- Probit regression
- The perceptron algorithm
- Support vector machines
- Linear discriminant analysis

Classifier setting:

- ✓ Based on Similarity
- ✓ Probabilistic Approach
- ✓ Decision-Boundary Approach
  - Geometric Approach

Classifiers Based on Similarity:

Once a good metric to define similarity, patterns can be classified by template matching or minimum distance classifier using a few prototypes per class.

The choice of the metric and prototypes is crucial to the success of this approach.

- ❖ Template Matching
  - Assign Pattern to the most similar template
- ❖ Nearest Mean Classifier
  - Assign Pattern to the nearest class mean
- ❖ Subspace Method
  - Assign Pattern to the nearest subspace (invariance)
- ❖ 1-Nearest Neighbor Rule
  - Assign Pattern to the class of the nearest training pattern

Probabilistic Approach:

<u>Bayes decision rule</u>

It takes into account costs associated with different types of misclassification.

Given prior probabilities, loss function, class-conditional densities, it is "optimal" in minimizing the risk.

Geometric Approach:

- Construct decision boundaries directly by optimizing certain error criterion.

Commonly used criterion

- Classification error
- MSE
- A training procedure is required.

With the 0/1 loss function, it assign a pattern to the class with the maximum posterior probability (maximum likelihood decision rule).

Classifier Learning:

Learning classifier systems can be split into two types depending upon where the genetic algorithm acts. A Pittsburgh-type LCS has a population of separate rule sets, where the genetic algorithm recombines and reproduces the best of these rule sets.

In a Michigan-style LCS there is only a single set of rules in a population and the algorithm's action focuses on selecting the best classifiers within that set. Michigan-style LCSs have two main types of fitness definitions, strength-based (e.g. **ZCS**) and accuracy-based (e.g. **XCS**). The term "learning classifier system" most often refers to Michigan-style LCSs.

Initially the classifiers or rules were binary, but recent research has expanded this representation to include real-valued, neural network, and functional (S-expression) conditions. Learning classifier systems are not fully understood mathematically and doing so remains an area of active research. Despite this, they have been successfully applied in many problem domains.

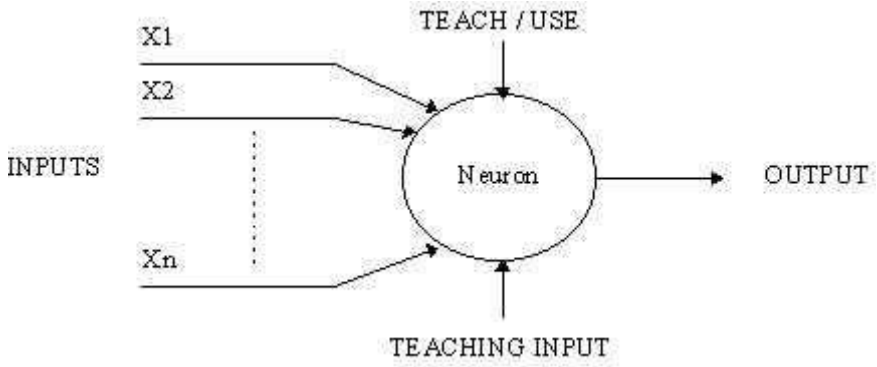
Cluster analysis:

- Grouping similar objects in a multidimensional space. It is useful for constructing new features which are abstractions of the existing features. Some algorithms, like k-means, simply partition the feature space.
- Other algorithms, like **single-link agglomeration**, create nested partitionings which form taxonomy. Another possibility is to learn a graph structure between the clusters, as in the Growing Neural Gas. The quality of the clustering depends crucially on the distance metric in the space.

NEURAL NETS:

An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to fire (or not), for particular input patterns.

In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.



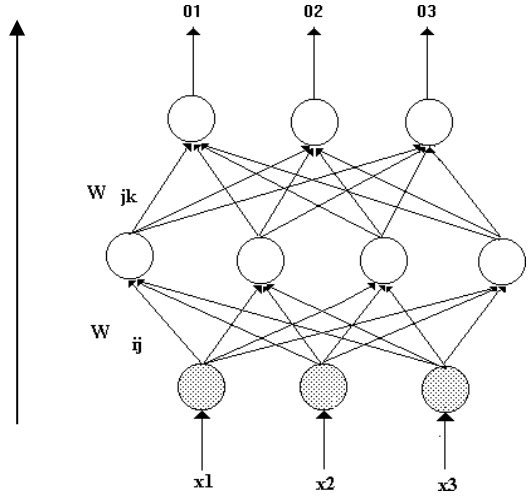
It computes the total weighted input xj, using the formula:

$$X_j = \sum_i y_i W_{ij}$$

Feed-forward networks:

Feed-forward ANNs allow signals to travel one way only; from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organisation is also referred to as bottom-up or top-down.

Output



Input

Unsupervised learning

It uses no external teacher and is based upon only local information. It is also referred to as self-organization, in the sense that it self-organizes data presented to the network and detects their emergent collective properties. Paradigms of unsupervised learning are Hebbian learning and competitive learning.

From Human Neurons to Artificial Neuron either aspect of learning concerns the distinction or not of a separate phase, during which the network is trained, or a subsequent operation phase. We say that a neural network learns off-line if the learning phase and the operation phase are distinct. A neural network learns on-line if it learns and operates at the same time. Usually, supervised learning is performed off-line, whereas unsupervised learning is performed on-line.

Fuzzy Systems:

Fuzzy systems are an alternative to traditional notions of set membership and logic that has its origins in ancient Greek philosophy, and applications at the leading edge of Artificial Intelligence.

It will present the foundations of fuzzy systems, along with some of the more noteworthy objections to its use, with examples drawn from current research in the field of Artificial Intelligence. Ultimately, it will be demonstrated that the use of fuzzy systems makes a viable addition to the field of Artificial Intelligence, and perhaps more generally to formal mathematics as a whole.

Basic Concepts

The notion central to fuzzy systems is that truth values (in fuzzy logic) or membership values (in fuzzy sets) are indicated by a value on the range [0.0, 1.0], with 0.0 representing absolute Falseness and 1.0 representing absolute Truth. For example, let us take the statement:

"Jane is old."

If Jane's age was 75, we might assign the statement the truth value of 0.80. The statement could be translated into set terminology as follows:

"Jane is a member of the set of old people."

This statement would be rendered symbolically with fuzzy sets as:

mOLD(Jane) = 0.80

Where m is the membership function, operating in this case on the fuzzy set of old people, which returns a value between 0.0 and 1.0.

At this juncture it is important to point out the distinction between fuzzy systems and probability. Both operate over the same numeric range, and at first glance both have similar values: 0.0 representing False (or non-membership), and 1.0 representing True (or membership). However, there is a distinction to be made between the two statements: The probabilistic approach yields the natural-language statement,

"There is an 80% chance that Jane is old," while the fuzzy terminology corresponds to "Jane's degree of membership within the set of old people is 0.80."

The semantic difference is significant: the first view supposes that Jane is or is not old (still caught in the Law of the Excluded Middle); it is just that we only have an 80% chance of knowing Šwhich set she is in. By contrast, fuzzy terminology supposes that Jane is "more or less" old, or some other term corresponding to the value of 0.80. Further distinctions arising out of the operations will be noted below.

The next step in establishing a complete system of fuzzy logic is to define the operations of EMPTY, EQUAL, COMPLEMENT (NOT), CONTAINMENT, UNION (OR), and INTERSECTION (AND). Before we can do this rigorously, we must state some formal definitions:

Definition 1: Let X be some set of objects, with elements noted as x. Thus,
X = {x}.

Definition 2: A fuzzy set A in X is characterized by a membership function mA(x) which maps each point in X onto the real interval [0.0, 1.0]. As mA(x) approaches 1.0, the "grade of membership" of x in A increases.

Definition 3: A is EMPTY iff for all x, mA(x) = 0.0.

Definition 4: A = B iff for all x: mA(x) = mB(x) [or, mA = mB].

Definition 5: mA' = 1 - mA.

Definition 6: A is CONTAINED in B iff mA <= mB.

Definition 7: C = A UNION B, where: mC(x) = MAX(mA(x), mB(x)).

Definition 8: C = A INTERSECTION B where: mC(x) = MIN(mA(x), mB(x)).

Fuzzy systems, including fuzzy logic and fuzzy set theory, provide a rich and meaningful addition to standard logic. The mathematics generated by these theories is consistent, and fuzzy logic may be a generalization of classic logic.

The applications which may be generated from or adapted to fuzzy logic are wide-ranging, and provide the opportunity for modeling of conditions which are inherently imprecisely defined, despite the concerns of classical logicians. Many systems may be modeled, simulated, and even replicated with the help of fuzzy systems, not the least of which is human reasoning itself.

MATHEMATICAL MORPHOLOGY

Mathematical morphology has been proved to be extremely useful in many image processing and analysis applications. In image processing, a well-known general approach is provided by mathematical morphology, where the images being analyzed are considered as sets of points and the set theory is applied

on the morphological operations. This approach is based upon logical relations between pixels, rather than arithmetic relations, and can extract geometric features by choosing a suitable structuring shape as a probe.

Mathematical morphology can extract image shape features, such as edges, fillets, holes, corners, wedges, and cracks, by operating with various shaped structuring elements (Maragos and Schafer, 1987b). In industrial vision applications, mathematical morphology can be used to implement fast object recognition, image enhancement, segmentation, and defect inspection.

In this chapter, we will introduce binary morphology, opening and closing, hit or- miss transform (HMT), grayscale morphology, basic morphological algorithms, and several variations of morphological filters, including alternating sequential filters (ASFs), recursive morphological filters, soft morphological filters, order-statistic soft morphological (OSSM) filters, recursive soft morphological filters, recursive order statistic soft morphological filters, regulated morphological filters, and fuzzy morphological filters.

Binary Morphology:

Mathematical morphology involves geometric analysis of shapes and textures in images. An image can be represented by a set of pixels. Morphological operators work with two images. The image being processed is referred to as the active image, and the other image, being a kernel, is referred to as the structuring element.

Each structuring element has a designed shape, which can be thought of as a probe or a filter of the active image. The active image can be modified by probing it with various structuring elements.

The elementary operations in mathematical morphology are dilation and erosion, which can be combined in sequence to produce other operations, such as opening and closing.

Binary Dilation:

The morphological transformation $\oplus$ combines two set using vector addition, e.g., (a,b)+(c,d)=(a+c+,b+d) the dilation $X \oplus B$ is the point set of all possible vector addition of pairs of elements, one from each of the sets X and B

$$X \oplus B = \{p \in \xi^{2:}\ p = x + b,\ x \in X \text{ and } b \in B\}$$

Binary Erosion:

The morphological transformation $\ominus$ combines two set using vector subtraction, the erosion $X \ominus B$ is the point set of all possible vector a subtraction of pairs of elements, one from each of the sets X and B

$$X \ominus B = \{p \in \xi^{2:}\ p = x + b,\ x \in X \text{ and } b \in B\}$$

Opening and closing:

In practical applications, dilation and erosion pairs are combined in sequence, either the dilation of an image followed by the erosion of the dilated result, or vice versa. In either case, the result of iteratively

applying dilations and erosions is an elimination of specific image details whose sizes are smaller than the structuring element without the global geometric distortion of unsuppressed features.

Erosion followed by dilation creates an important morphological transformation called opening. The opening of an image X by the structuring element B is denoted by $X \circ B$ and is defined as

$$X \circ B = (X \ominus B) \oplus B$$

Dilation followed by erosion creates an important morphological transformation called closing. The opening of an image X by the structuring element B is denoted by $X \bullet B$ and is defined as

$$X \bullet B = (X \oplus B) \ominus B$$

Hit-or-miss transform:

The hit-or-miss transformation on a binary image A is defined as follows: The structuring element B is a pair of binary images B1 and B2. The hit-or-miss transformation of A by (B1, B2), denoted by

$$X \otimes B = \{X: B1\ X \text{ and } B_2\ Xc\}$$

QUESTION BANK

5MARKS:

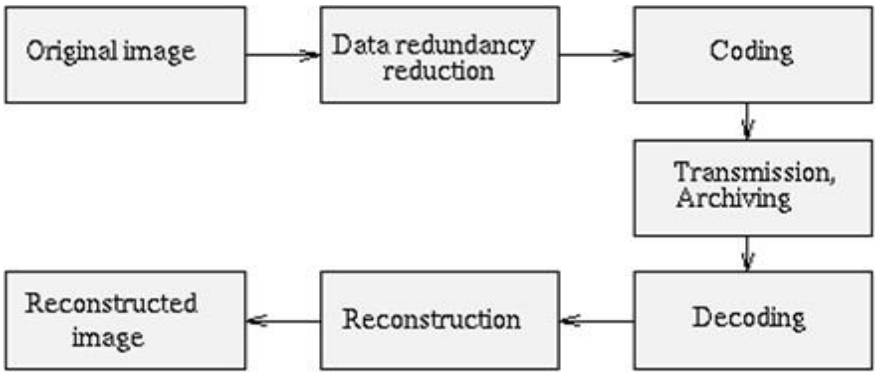
1. Explain the classifier setting in detail **(April 2012)**.
2. Explain the classifier learning in detail.
3. Explain the neural nets in detail **(April 2012)**.

10MARKS:

1. Explain the statistical pattern recognition in detail.
2. Explain the knowledge representation in detail **(April 2012)**..
3. Explain the mathematical morphology in detail.
4. Explain the fuzzy systems in detail **(April 2012)**.

IMAGE DATA COMPRESSION:

- Large amounts of data are used to represent an image.
- Technology permits ever-increasing image resolution (spatially and in gray levels), increasing numbers of spectral bands, and there is a consequent need to limit the resulting data volume.
- The amount of storage media needed is enormous. One possible approach to decreasing the necessary amount of storage is to work with compressed image data. Segmentation techniques have the side effect of image compression.
- If compression is the main goal of the algorithm, an image is represented using a lower number of bits per pixel, without losing the ability to reconstruct the image.
- It is necessary to find statistical properties of the image to design an appropriate compression transformation of the image; the more correlated the image data are, the more data items can be removed.



Data Compression and Image reconstruction

- Data compression methods can be divided into two principal groups:
- **Information preserving** compression permits error-free data reconstruction (**lossless compression**).
- Compression **methods with loss of information** do not preserve the information completely (**lossy compression**).
- Data compression success is usually measured in the reconstructed image by the mean squared error (MSE), signal to noise ratio etc. although these global error measures do not always reflect subjective image quality.
- Image data compression design - 2 parts.

  1) Image data properties determination
     - gray level histogram

- image entropy
- various correlation functions

2) Appropriate compression technique design.

IMAGE DATA PROPERTIES:

- **Entropy** - measure of image information content
- If an image has G gray levels, and the probability of gray level k is P(k), then entropy H_e, not considering correlation of gray levels, is defined as

$$H_e = -\sum_{k=0}^{G-1} P(k)\log_2(P(k)) \qquad (12.1)$$

- Information **redundancy** r is defined as

$$r = b - H_e \qquad (12.2)$$

- where b is the smallest number of bits with which the image quantization levels can be represented. A good estimate of entropy is usually not available. Image data entropy can however be estimated from a gray level histogram.
- Let h(k) be the frequency of gray level k in an image f, 0 <= k <= 2^b -1, and let the image size be M x N. The probability of occurrence of gray level k can be estimated as

$$\tilde{P}(k) = \frac{h(k)}{MN} \qquad (12.3)$$

and the entropy can be estimated as

$$\tilde{H}_e = -\sum_{k=0}^{2^b-1} \tilde{P}(k)\log_2(\tilde{P}(k)) \qquad (12.4)$$

- The information redundancy estimate is r = b- H_e. The definition of the **compression ratio** K is then

$$K = \frac{b}{\tilde{H}_e} \qquad (12.5)$$

- These formulae give theoretical limits of possible image compression.

DISCRETE IMAGE TRANSFORMS IN IMAGE DATA COMPRESSION:

- Basic idea: image data representation by coefficients of discrete image transforms
- The transform coefficients are ordered according to their importance
  - The least important (low contribution) coefficients are omitted.
- To remove correlated image data, the **Karhunen-Loeve transform** is the most important. This

transform builds a set of non-correlated variables with decreasing variance.

- The variance of a variable is a measure of its information content; therefore, a compression strategy is based on considering only transform variables with high variance, thus representing an image by only the first k coefficients of the transform.
- Other discrete image transforms discussed in the previous chapter are computationally less demanding. Cosine, Fourier, Hadamard, Walsh, or binary transforms are all suitable for image data compression.
- If an image is compressed using discrete transforms, it is usually divided into subimages of 8 x 8 or 16 x 16 pixels to speed up calculations, and then each subimage is transformed and processed separately.
- The same is true for image reconstruction, with each subimage being reconstructed and placed into the appropriate image position.

PREDICTIVE COMPRESSION METHODS:

- Predictive compressions use image information redundancy (correlation of data) to construct an estimate ~f(i,j) of the gray level value of an image element (i,j) from values of gray levels in the neighborhood of (i,j).
- In image parts where data are not correlated, the estimate ~f will not match the original value.
- The differences between estimates and reality, which may be expected to be relatively small in absolute terms, are coded and transmitted together with prediction model parameters -- the whole set now represents compressed image data.
- The gray value at the location (i,j) is reconstructed from a computed estimate ~f(i,j) and the stored difference d(i,j)

$$d(i,j) = \tilde{f}(i,j) - f(i,j) \qquad (12.6)$$

- Differential Pulse Code Modulation (DPCM)

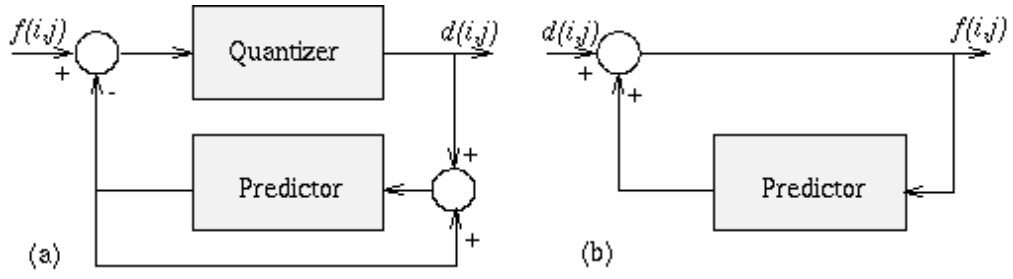


Figure 12.4 *Differential pulse code modulation: (a) Compression, (b) reconstruction.*

- Linear predictor of the third order is sufficient for estimation in a wide variety of images. The estimate ~f can be computed as

$$\tilde{f}(i,j) = a_1 f(i,j-1) + a_2 f(i-1,j-1) + a_3 f(i-1,j) \qquad (12.7)$$

- where a_1,a_2,a_3 are image prediction model parameters. These parameters are set to minimize the mean quadratic estimation error e,

$$e = \mathcal{E}([\tilde{f}(i,j) - f(i,j)]^2) \qquad (12.8)$$

- and the solution, assuming f is a stationary random process with a zero mean, using a predictor of the third order, is

$$\begin{aligned} a_1 R(0,0) + a_2 R(1,0) + a_3 R(1,1) &= R(0,1) \\ a_1 R(1,0) + a_2 R(0,0) + a_3 R(0,1) &= R(1,1) \\ a_1 R(1,1) + a_2 R(0,1) + a_3 R(0,0) &= R(1,0) \end{aligned} \qquad (12.9)$$

- where R(m,n) is the autocorrelation function of the random process f.

HIERARCHICAL AND PROGRESSIVE COMPRESSION TECHNIQUES:

- A substantial reduction in bit volume can be obtained by merely representing a source as a pyramid.
- Approaches exist for which the entire pyramid requires data volume equal to that of the full resolution image.
- Even more significant reduction can be achieved for images with large areas of the same gray level if a quadtree coding scheme is applied.

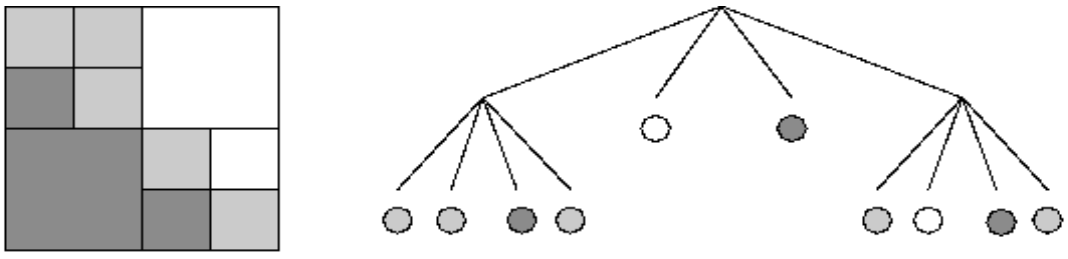


Figure 12.6 *Principle of quadtree image compression; original image and the corresponding quadtree.*

- **Progressive image transmission** - transmitting all image data may not be necessary under some circumstances - e.g., searching an image database looking for a particular image.
- This approach is also commonly used to decrease the waiting time needed for the image to start appearing after transmission and is used by World Wide Web image transmissions.

- In progressive transmission, the images are represented in a pyramid structure, the higher pyramid levels (lower resolution) being transmitted first.
- The concept of **smart compression** is based on the sensing properties of human visual sensors.
- The spatial resolution of the human eye decreases significantly with increasing distance from the optical axis.
- The main difficulty remains in determining the areas of interest in the image on which the user will focus.
- When considering a **smart progressive image transmission**, the image should be transmitted in higher resolution in areas of interest first - this improves a subjective rating of transmission speed as sensed by a human user.
- This smart image transmission and compression may be extremely useful if applied to dynamic image generators in driving or flight simulators, or to high definition television.

COMPARISON OF COMPRESSION METHODS:

- Transform-based methods better preserve subjective image quality, and are less sensitive to statistical image property changes both inside a single image and between images.
- Prediction methods, on the other hand, can achieve larger compression ratios in a much less expensive way, tend to be much faster than transform-based or vector quantization compression schemes, and can easily be realized in hardware.
- If compressed images are transmitted, an important property is insensitivity to transmission channel noise. Transform-based techniques are significantly less sensitive to the channel noise - if a transform coefficient is corrupted during transmission, the resulting image distortion is homogeneously spread through the image or image part and is not too disturbing.
- Pyramid based techniques have a natural compression ability and show a potential for further improvement of compression ratios. They are suitable for dynamic image compression and for progressive and smart transmission approaches.

CODING:

- Very well known is **Huffman encoding** which can provide optimal compression and error-free decompression.
- The main idea of Huffman coding is to represent data by codes of variable length, with more frequent data being represented by shorter codes.
- Many modifications of the original algorithm exist, with recent adaptive Huffman coding algorithms requiring only one pass over the data.

- More recently, the **Lempel-Ziv (or Lempel-Ziv-Welch, LZW) algorithm** of **dictionary-based coding** has found wide favor as a standard compression algorithm.
- In this approach, data are represented by pointers referring to a dictionary of symbols.
- These, and a number of similar techniques, are in widespread use for de-facto standard image representations which are popular for Internet and World Wide Web image exchange.
    - **GIF** format (Graphics Interchange Format) is probably the most popular currently in use.
        - GIF is a creation of Compuserve Inc., and is designed for the encoding of RGB images (and the appropriate palette with pixel depths between 1 and 8 bits.
        - Blocks of data are encoded using the LZW algorithm.
        - GIF has two versions - 87a and 89a, the latter supporting the storing of text and graphics in the same file.
    - **TIFF** (Tagged Image File Format) was first defined by the Aldus Corporation in 1986, and has gone through a number of versions to incorporate RGB color, compressed color (LZW), other color formats and ultimately (in Version 6), JPEG compression (below) -- these versions all have backward compatibility.

JPEG AND MPEG IMAGE COMPRESSION:

- There is an increasing effort to achieve standardization in image compression.
- The Joint Photographic Experts Group (**JPEG**) has developed an international standard for general purpose, color, still-image compression.
- **MPEG** standard (Motion Picture Experts Group) was developed for full-motion video image sequences with applications to digital video distribution and high definition television (HDTV) in mind.

JPEG - still image compression:

- JPEG is widely used in many application areas. Four compression modes are furnished
    1. Sequential DCT-based compression
    2. Progressive DCT-based compression
    3. Sequential lossless predictive compression
    4. Hierarchical lossy or lossless compression
- While the lossy compression modes were designed to achieve compression ratios around 15 with very good or excellent image quality, the quality deteriorates for higher compression ratios. A compression ratio between 2 and 3 is typically achieved in the lossless mode.

Sequential JPEG Compression:

- Consists of a forward DCT transform, a quantizer, and entropy encoder while decompression starts with entropy decoding followed by dequantizing and inverse DCT.

- In the compression stage, the unsigned image values from the interval [0,(2^b)-1] are first shifted to cover the interval [-2^(b-1),2^(b-1)-1].
- The image is then divided into 8x8 blocks and each block is independently transformed into the frequency domain using the DCT-II transform.
- Many of the 64 DCT coefficients have zero or near-zero values in typical 8x8 blocks which forms the basis for compression.
- The 64 coefficients are quantized using a quantization table Q(u,v) of integers from 0 to 255 that is specified by the application to reduce the storage/transmission requirements of coefficients that contribute little or nothing to the image content.
    - The following formula is used for quantization

$$F_Q(u,v) = round\left[\frac{F(u,v)}{Q(u,v}\right] \qquad (12.10)$$

- After quantization, the dc-coefficient F(0,0) is followed by the 63 ac-coefficients that are ordered in a 2D matrix in a zig-zag fashion according to their increasing frequency.
- The dc-coefficients are then encoded using predictive coding, the rationale being that average gray levels of adjacent 8x8 blocks (dc-coefficients) tend to be similar.
- The last step of the sequential JPEG compression algorithm is entropy encoding.
    - Two approaches are specified by the JPEG standard.
    - The baseline system uses simple Huffman coding while the extended system uses arithmetic coding and is suitable for a wider range of applications.
- Sequential JPEG decompression uses all the steps described above in the reverse order. After entropy decoding (Huffman or arithmetic), the symbols are converted into DCT coefficients and dequantized

$$F'_Q(u,v) = F_Q(u,v)Q(u,v) \qquad (12.11)$$

- where again, the Q(u,v) are quantization coefficients from the quantization table that is transmitted together with the image data.
- Finally, the inverse DCT transform is performed and the image gray values are shifted back to the interval [0,(2^b)-1].

Progressive JPEG Compression:

A sequence of scans is produced, each scan containing a coded subset of DCT coefficients.

- A buffer is needed at the output of the quantizer to store all DCT coefficients of the entire image.
- These coefficients are selectively encoded.

- Three algorithms are defined as part of the JPEG progressive compression standard;
    - **progressive spectral selection**
    - **progressive successive approximation**
    - **Combined progressive algorithm**.
- In the progressive spectral selection approach, the dc-coefficients are transmitted first, followed by groups of low frequency and higher frequency coefficients.
- In the progressive successive approximation, all DCT coefficients are sent first with lower precision and their precision increases as additional scans are transmitted.
- The combined progressive algorithm uses both of the above principles together.

Sequential Lossless JPEG Compression:

- The lossless mode of the JPEG compression uses a simple predictive compression algorithm and Huffman coding to encode the prediction differences.

Hierarchical JPEG Compression:

- Using the hierarchical JPEG mode, decoded images can be displayed either progressively or at different resolutions.
- A pyramid of images is created and each lower resolution image is used as a prediction for the next higher resolution pyramid level.
- The three main JPEG modes can be used to encode the lower-resolution images - sequential DCT, progressive DCT, or lossless.
- In addition to still image JPEG compression, motion JPEG (**MJPEG**) compression exists that can be applied to real-time full-motion applications.
- However, MPEG compression represents a more common standard and is described below.

QUESTION BANK

5 MARKS:

1. Explain the image data properties in detail **(April 2012)**.
2. Explain the predictive compression methods in detail.
3. Explain the comparison of compression methods in detail **(April 2012)**.

10 MARKS:

1. Explain the image data compression in detail.
2. Explain the discrete image transforms in image data compression in detail **(April 2012)**.
3. Explain the hierarchical and progressive compression techniques in detail.
4. Explain the JPEG and MPEG image compression in detail **(April 2012)**.

COMPLETE REFERENCE

1. Rafael C. Gonzalez, Richard E.Woods, Digital Image Processing, Prentice Hall, Third Edition, 2008.
2. Sonka, Hlavac, Boyle, Digital Image Processing and Computer Vision, Cengage Learning, 2009.
3. Anil.K.Jain, Fundamentals of Digital Image Processing, Prentice-Hall, 1989.
4. Chanda & Majumdar, Digital Image Processing and Analysis, Prentice Hall ,3rd Edition.
5. http://www.engineering.uiowa.edu/~dip/lecture/lecture.html
6. http://www.intelligence.tuc/~petrakis
7. http://www.research.microsoft.com/~szeliski/Book
