1. What is meant by Digital Image Processing? Explain how digital images can be represented?

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 amplitude 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 referred to as picture elements, image elements, pels, and pixels. Pixel is the term most widely used 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 ultra-sound, electron microscopy, and computer-generated images. Thus, digital image processing encompasses a wide and varied field of applications. There is no general agreement among authors regarding where image processing stops and other related areas, such as image analysis and computer vision, start. Sometimes a distinction is made by defining image processing as a discipline in which both the input and output of a process are images. We believe this to be a limiting and somewhat artificial boundary. 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-, mid-, and 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). Finally, higher-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 and, in addition, encompasses processes that extract attributes from images, up to and including the recognition of individual objects. As a simple illustration to clarify these concepts, consider the area of automated analysis of text. The processes of acquiring an image of the area containing the text, preprocessing that image, extracting (segmenting) the individual characters, describing the characters in a form suitable for computer processing, and recognizing those individual characters are in the scope of what we call digital image processing.

Representing Digital Images:

We will use two principal ways to represent digital images. Assume that an image f(x, y) is sampled so that the resulting digital image has M rows and N columns. The values of the coordinates (x, y) now become discrete quantities. For notational clarity and convenience, we shall use integer values for these discrete coordinates. Thus, the values of the coordinates at the origin are (x, y) = (0, 0). The next coordinate values along the first row of the image are represented as (x, y) = (0, 1). It is important to keep in mind that the notation (0, 1) is used to signify the second sample along the first row. It does not mean that these are the actual values of physical coordinates when the image was sampled. Figure 1 shows the coordinate convention used.

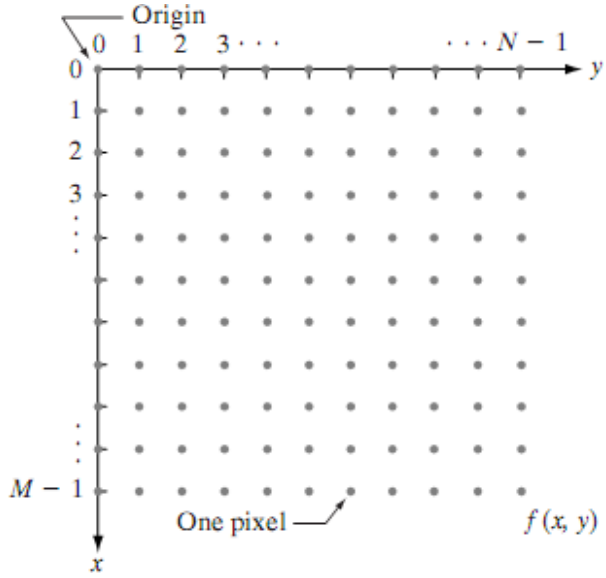


Fig 1 Coordinate convention used to represent digital images

The notation introduced in the preceding paragraph allows us to write the complete M*N digital image in the following compact matrix form:

$$f(x, y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}.$$

The right side of this equation is by definition a digital image. Each element of this matrix array is called an image element, picture element, pixel, or pel.

2. What are the fundamental steps in Digital Image Processing?

Fundamental Steps in Digital Image Processing:

Image acquisition is the first process shown in Fig.2. Note that 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 among the simplest and most appealing areas of digital image processing. Basically, the idea behind enhancement techniques is to bring out detail that is obscured, or simply to highlight certain features of interest in an image. A familiar example of enhancement is when we increase the contrast of an image because "it looks better." It is important to keep in mind that enhancement is a very subjective area of image processing.

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.

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.

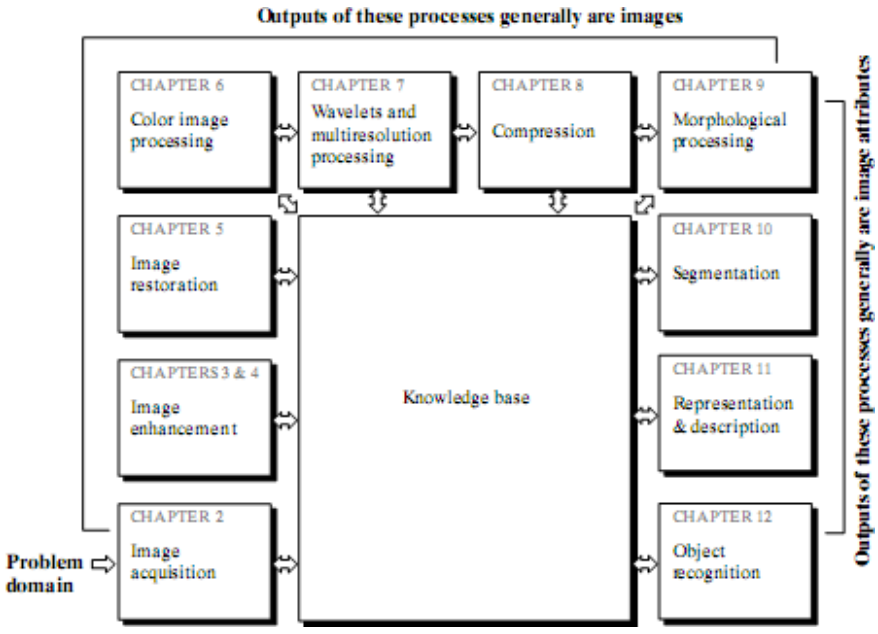


Fig.2. Fundamental steps in Digital Image Processing

Wavelets are the foundation for representing images in various degrees of resolution. Compression, as the name implies, deals with techniques for reducing the storage required to save an image, or the bandwidth required to transmit 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.

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. We conclude our coverage of digital image processing with the development of methods for recognition of individual objects.

3. What are the components of an Image Processing System?

Components of an Image Processing System:

As recently as the mid-1980s, numerous models of image processing systems being sold throughout the world were rather substantial peripheral devices that attached to equally substantial host computers. Late in the 1980s and early in the 1990s, the market shifted to image processing hardware in the form of single boards designed to be compatible with industry standard buses and to fit into engineering workstation cabinets and personal computers. In addition to lowering costs, this market shift also served as a catalyst for a significant number of new companies whose specialty is the development of software written specifically for image processing.

Although large-scale image processing systems still are being sold for massive imaging applications, such as processing of satellite images, the trend continues toward miniaturizing and blending of general-purpose small computers with specialized image processing hardware. Figure 3 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), which 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 framess) that the typical main computer cannot handle.

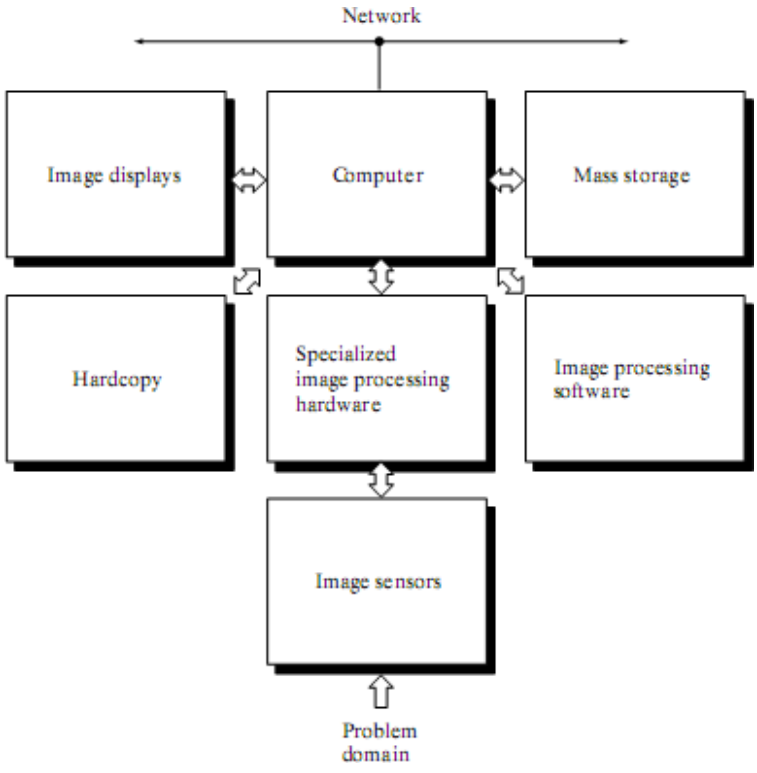


Fig.3. Components of a general purpose Image Processing System

The computer in an image processing system is a general-purpose computer and can range from a PC to a supercomputer. In dedicated applications, some times specially designed 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 offline 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*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 re-call, and (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 shown in Fig.3.Online 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. Magnetic tapes and optical disks housed in “jukeboxes” are the usual media for archival applications.

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. In some cases, it is necessary to have stereo displays, and these are implemented in the form of headgear containing two small displays embedded in goggles worn by the user.

Hardcopy devices for recording images include laser printers, film cameras, heat-sensitive devices, inkjet units, and digital units, such as optical and CD-ROM 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. Fortunately, this situation is improving quickly as a result of optical fiber and other broadband technologies.

4. Explain about elements of visual perception.

Elements of Visual Perception:

Although the digital image processing field 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.

(1) Structure of the Human Eye:

Figure 4.1 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 optical globe. At its anterior extreme, the choroid is divided into the ciliary body and the iris diaphragm. 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.

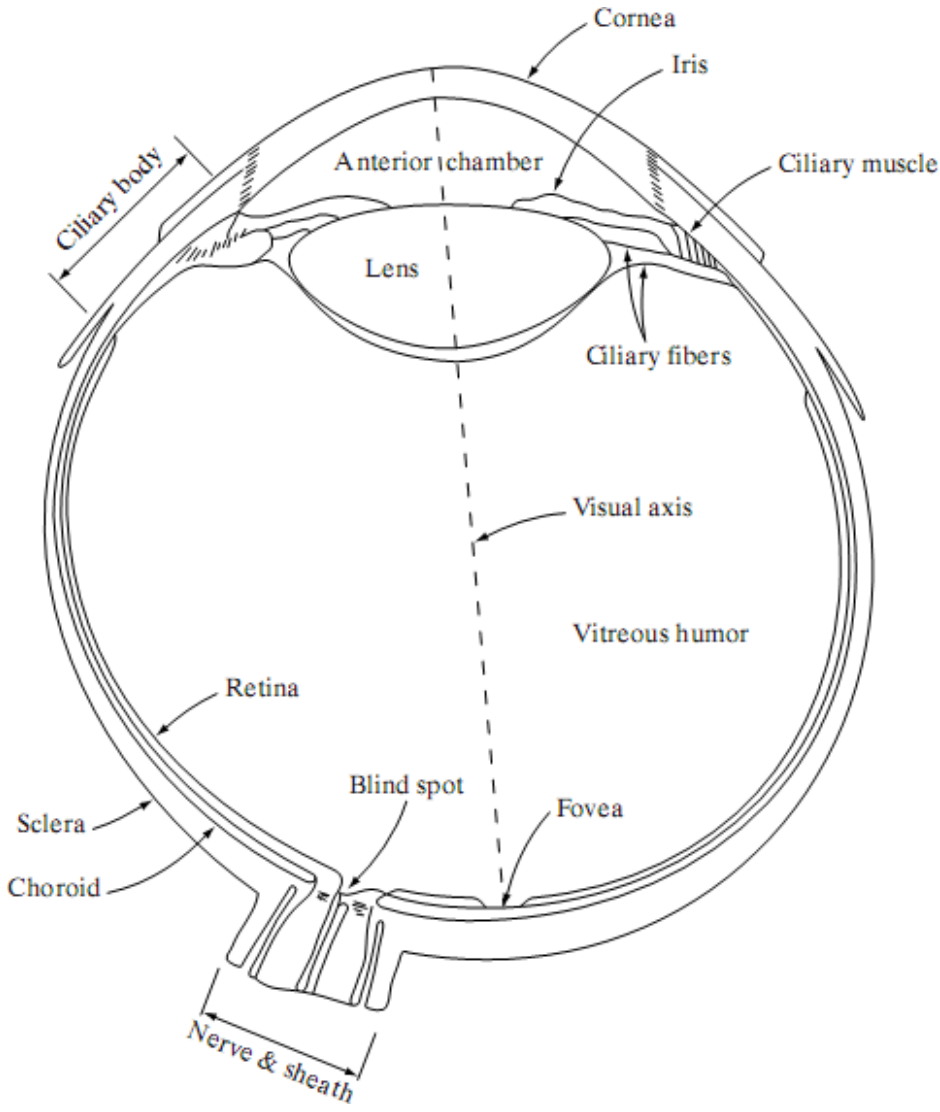


Fig.4.1 Simplified diagram of a cross section of the human eye.

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. The number of rods is much larger: Some 75 to 150 million are distributed over the retinal surface. The larger area of distribution and the fact that several rods are connected to a single nerve end reduce the amount of detail discernible by these receptors. Rods serve to give a general, overall picture of the field of view. They are not involved in color vision and are sensitive to low levels of illumination. For example, objects that appear brightly colored in daylight when seen by moonlight appear as colorless forms because only the rods are stimulated. This phenomenon is known as scotopic or dim-light vision.

(2) Image Formation in the Eye:

The principal difference between the lens of the eye and an ordinary optical lens is that the former is flexible. As illustrated in Fig. 4.1, the radius of curvature of the anterior surface of the lens is greater than the radius of its posterior surface. The shape of the lens is controlled by tension in the fibers of the ciliary body. To focus on distant objects, the controlling muscles cause the lens to be relatively flattened. Similarly, these muscles allow the lens to become thicker in order to focus on objects near the eye. The distance between the center of the lens and the retina (called the focal length) varies from approximately 17 mm to about 14 mm, as the refractive power of the lens increases from its minimum to its maximum. When the eye

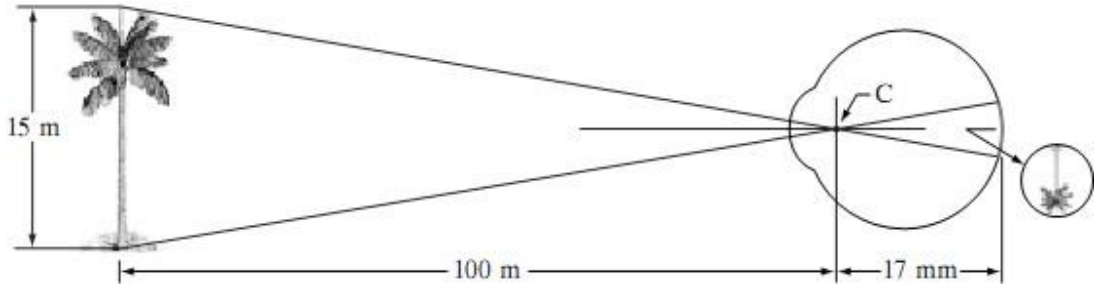


Fig.4.2. Graphical representation of the eye looking at a palm tree Point C is the optical center of the lens.

focuses on an object farther away than about 3 m, the lens exhibits its lowest refractive power. When the eye focuses on a nearby object, the lens is most strongly refractive. This information

makes it easy to calculate the size of the retinal image of any object. In Fig. 4.2, for example, the observer is looking at a tree 15 m high at a distance of 100 m. If h is the height in mm of that object in the retinal image, the geometry of Fig.4.2 yields 15/100=h/17 or h=2.55mm. The retinal image is reflected primarily in the area of the fovea. Perception then takes place by the relative excitation of light receptors, which transform radiant energy into electrical impulses that are ultimately decoded by the brain.

(3)Brightness Adaptation and Discrimination:

Because digital images are displayed as a discrete set of intensities, the eye's ability to discriminate between different intensity levels is an important consideration in presenting image-processing results. The range of light intensity levels to which the human visual system can adapt is enormous—on the order of 10^{10}—from the scotopic threshold to the glare limit. Experimental evidence indicates that subjective brightness (intensity as perceived by the human visual system) is a logarithmic function of the light intensity incident on the eye. Figure 4.3, a plot of light intensity versus subjective brightness, illustrates this characteristic. The long solid curve represents the range of intensities to which the visual system can adapt. In photopic vision alone, the range is about 10^6. The transition from scotopic to photopic vision is gradual over the approximate range from 0.001 to 0.1 millilambert (–3 to –1 mL in the log scale), as the double branches of the adaptation curve in this range show.

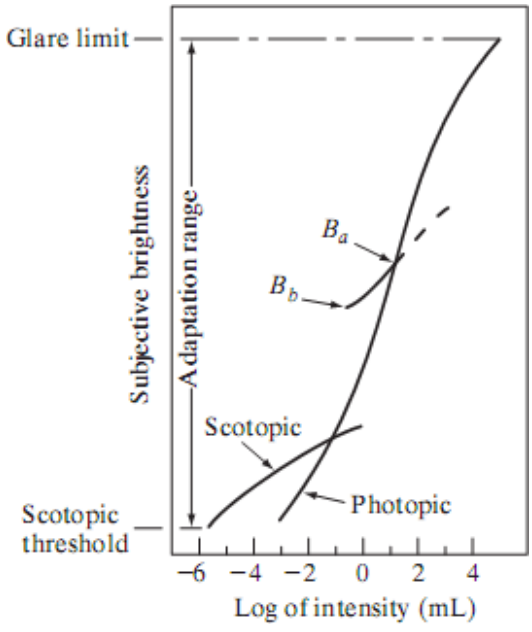


Fig.4.3. Range of Subjective brightness sensations showing a particular adaptation level.

The essential point in interpreting the impressive dynamic range depicted in Fig.4.3 is that the visual system cannot operate over such a range simultaneously. Rather, it accomplishes this large variation by changes in its overall sensitivity, a phenomenon known as brightness adaptation. The total range of distinct intensity levels it can discriminate simultaneously is rather small when compared with the total adaptation range. For any given set of conditions, the current sensitivity level of the visual system is called the brightness adaptation level, which may correspond, for example, to brightness Ba in Fig. 4.3. The short intersecting curve represents the range of subjective brightness that the eye can perceive when adapted to this level. This range is rather restricted, having a level Bb at and below which all stimuli are perceived as indistinguishable blacks. The upper (dashed) portion of the curve is not actually restricted but, if extended too far, loses its meaning because much higher intensities would simply raise the adaptation level higher than Ba.

5. Explain the process of image acquisition.

Image Sensing and Acquisition:

The types of 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. We enclose illumination and scene in quotes to emphasize the fact that they are considerably more general than the familiar situation in which a visible light source illuminates a common everyday 3-D (three-dimensional) scene. For example, the illumination may originate from a source of electromagnetic energy such as radar, infrared, or X-ray energy. But, as noted earlier, it could originate from less traditional sources, such as ultrasound or even a computer-generated illumination pattern.

Similarly, the scene elements could be familiar objects, but they can just as easily be molecules, buried rock formations, or a human brain. We could even image a source, such as acquiring images of the sun. Depending on the nature of the source, illumination energy is reflected from, or transmitted through, objects. 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 5.1 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.

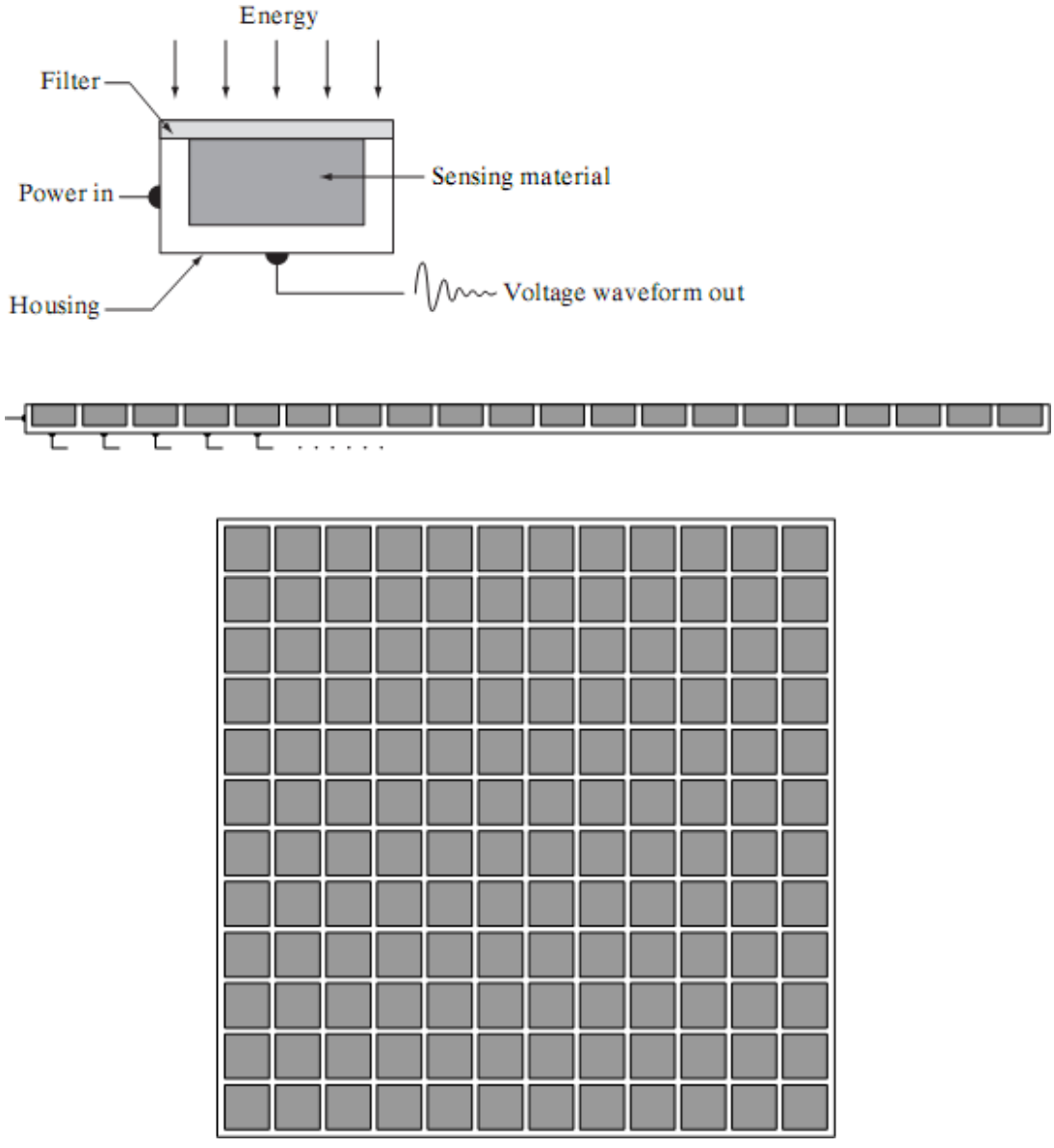


Fig.5.1 (a) Single imaging Sensor (b) Line sensor (c) Array sensor

(1) Image Acquisition Using a Single Sensor:

Figure 5.1 (a) shows the components of a single sensor. Perhaps the most familiar sensor of this type is the photodiode, which is constructed of silicon materials and whose output voltage

waveform is proportional to light. The use of a filter in front of a sensor improves selectivity. 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. Figure 5.2 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. Since 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.

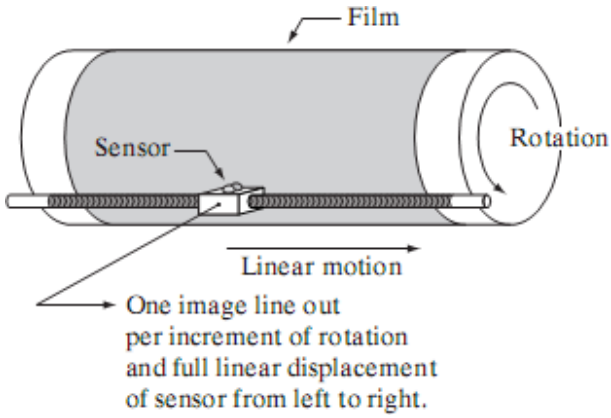


Fig.5.2. Combining a single sensor with motion to generate a 2-D image

(2) Image Acquisition Using Sensor Strips:

A geometry that is used much more frequently than single sensors consists of an in-line arrangement of sensors in the form of a sensor strip, as Fig. 5.1 (b) shows. The strip provides imaging elements in one direction. Motion perpendicular to the strip provides imaging in the other direction, as shown in Fig. 5.3 (a).This is the type of arrangement used in most flat bed scanners. 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. 5.3 (b) shows. A rotating X-ray source provides illumination and the portion of the sensors opposite the source collect the X-ray energy that pass 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. 5.3 (b).

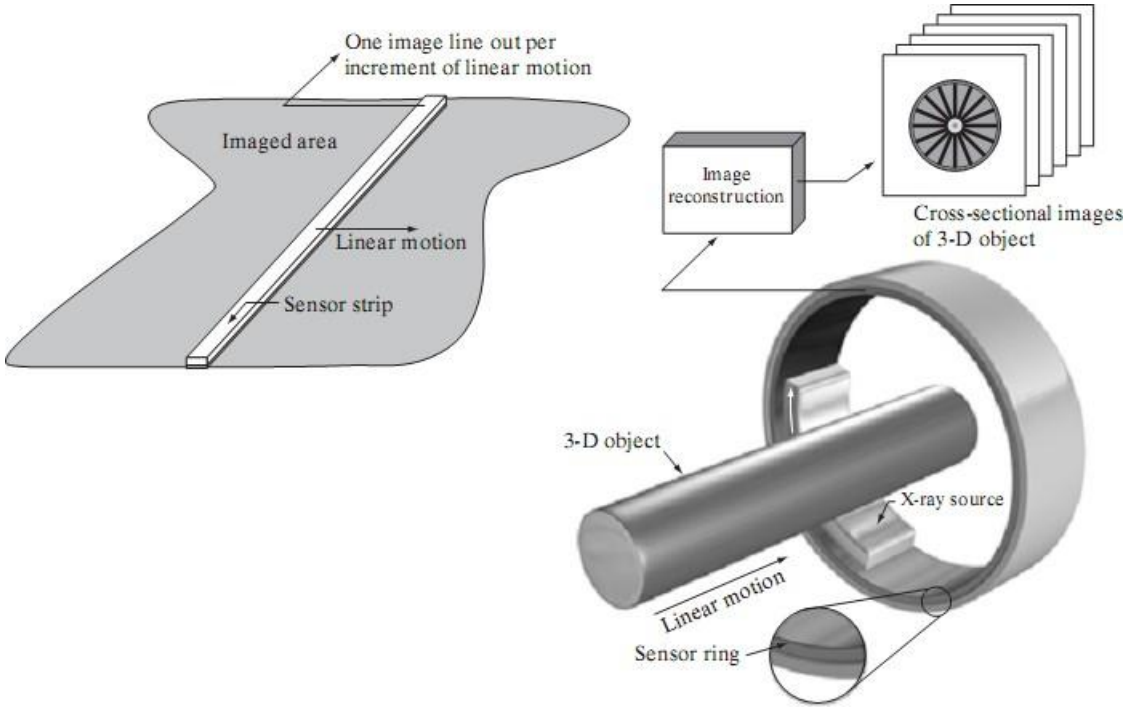
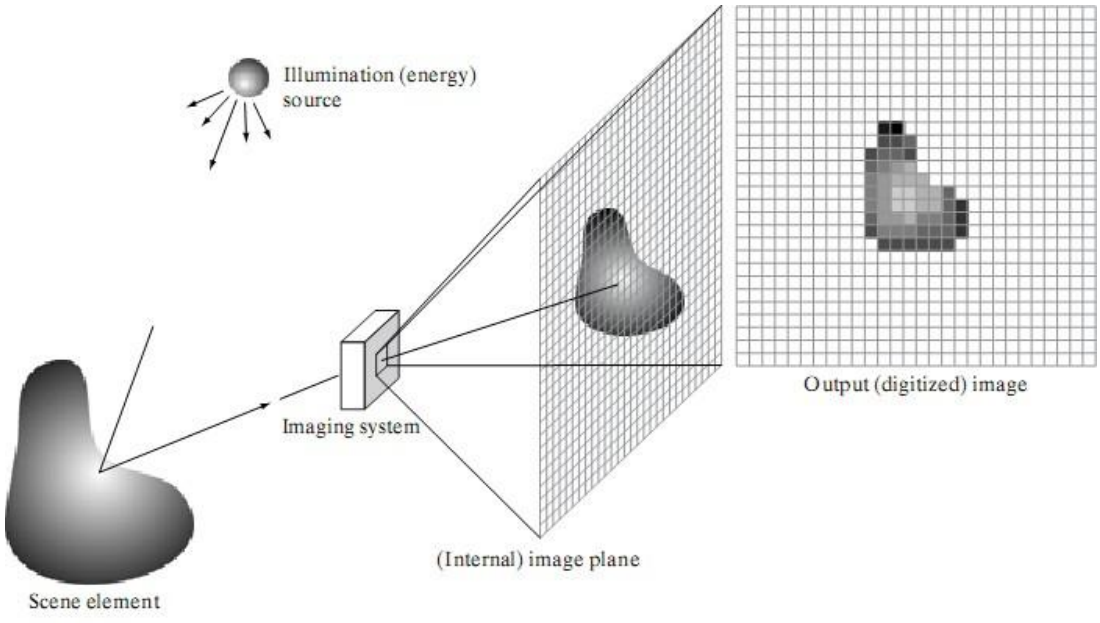


Fig.5.3 (a) Image acquisition using a linear sensor strip (b) Image acquisition using a

circular sensor strip.

(3) Image Acquisition Using Sensor Arrays:

Figure 5.1 (c) shows individual sensors arranged in the form of a 2-D array. 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 4000 * 4000 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. Since the sensor array shown in Fig. 5.4 (c) 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. The principal manner in which array sensors are used is shown in Fig.5.4. This figure shows the energy from an illumination source being reflected from a scene element, but, as mentioned at the beginning of this section, the energy also could be transmitted through the scene elements. The first function performed by the imaging system shown in Fig.5.4 (c) is to collect the incoming energy and focus it onto an image plane. If the illumination is light, the front end of the imaging system is a lens, which projects the viewed scene onto the lens focal plane, as Fig. 2.15(d) shows. The sensor array, which is coincident with the focal plane, produces outputs proportional to the integral of the light received at each sensor. Digital and analog circuitry sweep these outputs and converts them to a video signal, which is then digitized by another section of the imaging system. The output is a digital image, as shown diagrammatically in Fig. 5.4 (e).



a b c d e

Fig.5.4 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

6. Explain about image sampling and quantization process.

Image Sampling and Quantization:

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 Fig.6.1. Figure 6.1(a) shows a continuous image, f(x, y), 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 shown in Fig.6.1 (b) is a plot of amplitude (gray level) values of the continuous image along the line segment AB in Fig. 6.1(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.6.1 (c).The location of each sample is given 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 gray-level values. In order to form a digital function, the gray-level values also must be converted (quantized) into discrete quantities. The right side of Fig. 6.1 (c) shows the gray-level scale divided into eight discrete levels, ranging from black to white. The vertical tick marks indicate the specific value assigned to each of the eight gray levels. The continuous gray levels are quantized simply by assigning one of the eight discrete gray levels 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.6.1 (d). Starting at the top of the image and carrying out this procedure line by line produces a two-dimensional digital image.

Sampling in the manner just described assumes that we have a continuous image in both coordinate directions as well as in amplitude. In practice, the method of sampling is determined by the sensor arrangement used to generate the image. When an image is generated by a single

sensing element combined with mechanical motion, as in Fig. 2.13, the output of the sensor is quantized in the manner described above. However, sampling is accomplished by selecting the number of individual mechanical increments at which we activate the sensor to collect data. Mechanical motion can be made very exact so, in principle; there is almost no limit as to how fine we can sample an image. However, practical limits are established by imperfections in the optics used to focus on the

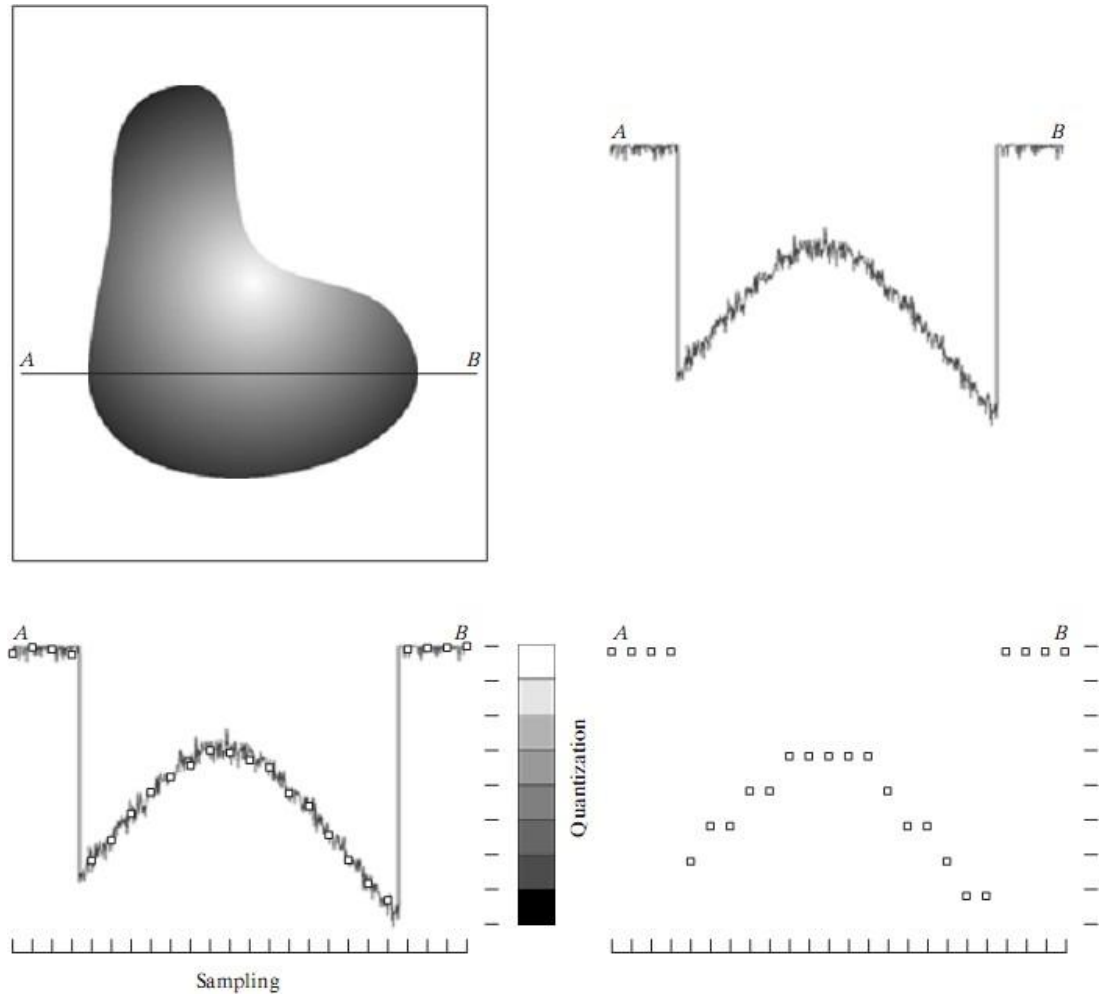
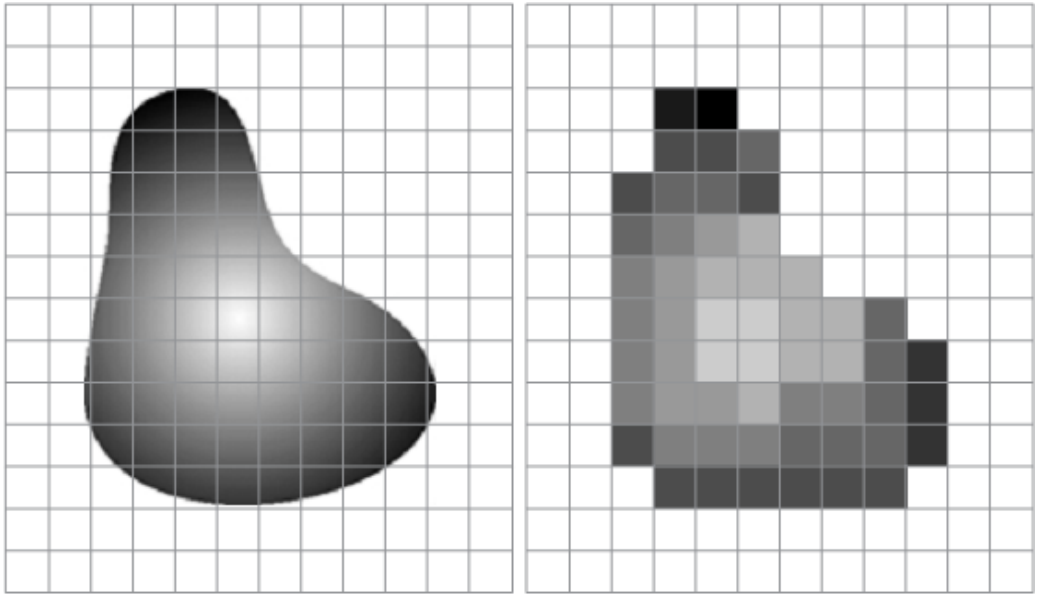


Fig.6.1. Generating a digital image (a) Continuous image (b) A scan line from A to Bin the continuous image, used to illustrate the concepts of sampling and quantization (c) Sampling and quantization. (d) Digital scan line

sensor an illumination spot that is inconsistent with the fine resolution achievable with mechanical displacements. When a sensing strip is used for image acquisition, the number of sensors in the strip establishes the sampling limitations in one image direction. Mechanical motion in the other direction can be controlled more accurately, but it makes little sense to try to achieve sampling density in one direction that exceeds the sampling limits established by the number of sensors in the other. Quantization of the sensor outputs completes the process of generating a digital image.

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. Figure 6.2 illustrates this concept. Figure 6.2 (a) shows a continuous image projected onto the plane of an array sensor. Figure 6.2 (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 gray levels used in sampling and quantization.



a b

Fig.6.2. (a) Continuos image projected onto a sensor array (b) Result of image sampling and quantization.

7. Define spatial and gray level resolution. Explain about isopreference curves.

Spatial and Gray-Level Resolution:

Sampling is the principal factor determining the spatial resolution of an image. Basically, spatial resolution is the smallest discernible detail in an image. Suppose that we construct a chart with vertical lines of width W, with the space between the lines also having width W.A line pair consists of one such line and its adjacent space. Thus, the width of a line pair is 2W, and there are 1/2Wline pairs per unit distance. A widely used definition of resolution is simply the smallest number of discernible line pairs per unit distance; for example, 100 line pairs per millimeter. Gray-level resolution similarly refers to the smallest discernible change in gray level. We have considerable discretion regarding the number of samples used to generate a digital image, but this is not true for the number of gray levels. Due to hardware considerations, the number of gray levels is usually an integer power of 2.

The most common number is 8 bits, with 16 bits being used in some applications where enhancement of specific gray-level ranges is necessary. Sometimes we find systems that can digitize the gray levels of an image with 10 or 12 bit of accuracy, but these are the exception rather than the rule. When an actual measure of physical resolution relating pixels and the level of detail they resolve in the original scene are not necessary, it is not uncommon to refer to an L-level digital image of size M*N as having a spatial resolution of M*N pixels and a gray-level resolution of L levels.

Fig.7.1. A 1024*1024, 8-bit image subsampled down to size 32*32 pixels The number of allowable gray levels was kept at 256.

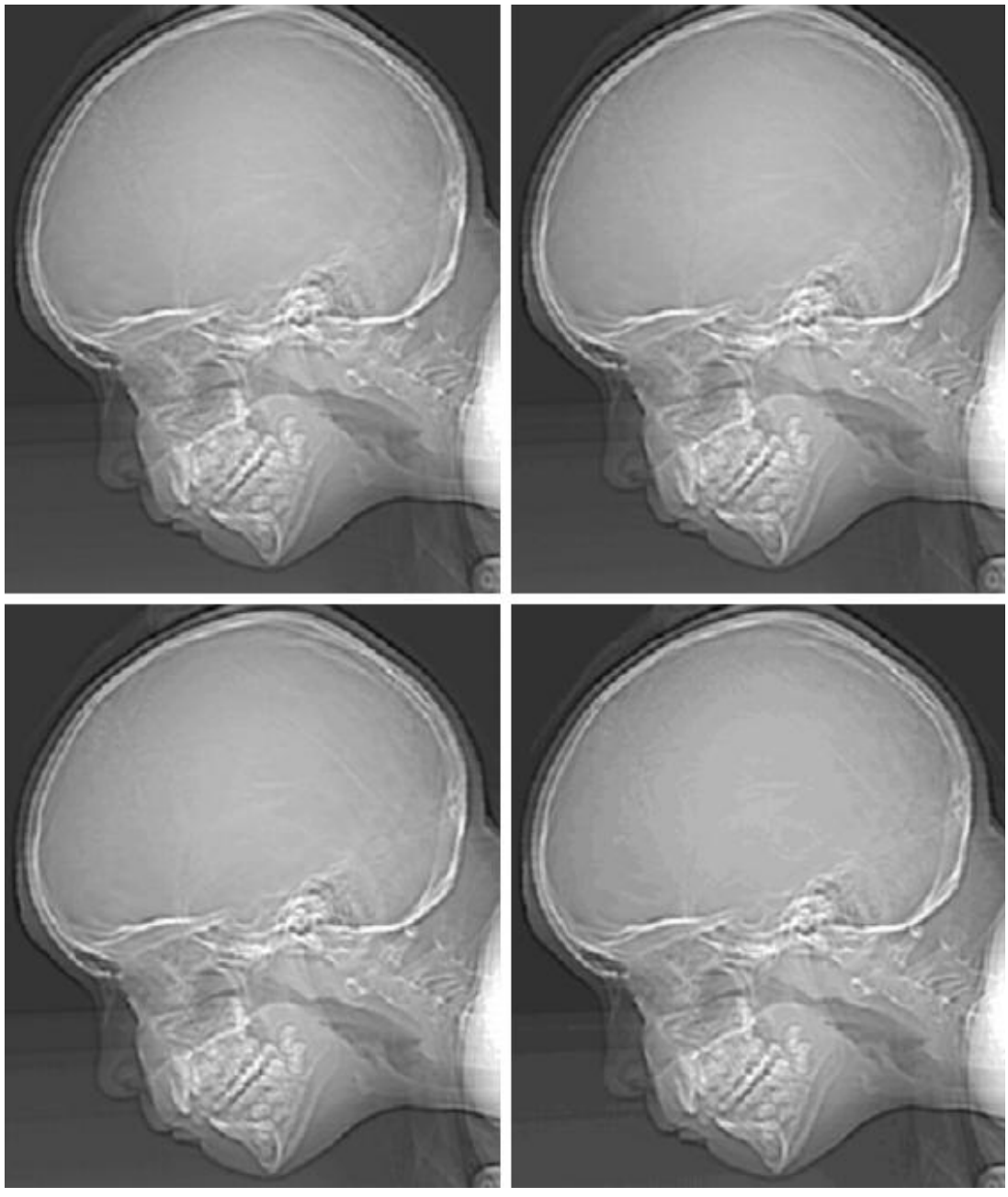
The subsampling was accomplished by deleting the appropriate number of rows and columns from the original image. For example, the 512*512 image was obtained by deleting every other row and column from the 1024*1024 image. The 256*256 image was generated by deleting every other row and column in the 512*512 image, and so on. The number of allowed gray levels was kept at 256. These images show the dimensional proportions between various sampling densities, but their size differences make it difficult to see the effects resulting from a reduction in the number of samples. The simplest way to compare these effects is to bring all the subsampled images up to size 1024*1024 by row and column pixel replication. The results are shown in Figs. 7.2 (b) through (f). Figure7.2 (a) is the same 1024*1024, 256-level image shown in Fig.7.1; it is repeated to facilitate comparisons.

a b c
d e f

Fig. 7.2 (a) 1024*1024, 8-bit image (b) 512*512 image resampled into 1024*1024 pixels by row and column duplication (c) through (f) 256*256, 128*128, 64*64, and 32*32 images resampled into 1024*1024 pixels

Compare Fig. 7.2(a) with the 512*512 image in Fig. 7.2(b) and note that it is virtually impossible to tell these two images apart. The level of detail lost is simply too fine to be seen on the printed page at the scale in which these images are shown. Next, the 256*256 image in Fig. 7.2(c) shows a very slight fine checkerboard pattern in the borders between flower petals and the black background. A slightly more pronounced graininess throughout the image also is beginning to appear. These effects are much more visible in the 128*128 image in Fig. 7.2(d), and they become pronounced in the 64*64 and 32*32 images in Figs. 7.2 (e) and (f), respectively.

In the next example, we keep the number of samples constant and reduce the number of gray levels from 256 to 2, in integer powers of 2.Figure 7.3(a) is a 452*374 CAT projection image, displayed with k=8 (256 gray levels). Images such as this are obtained by fixing the X-ray source in one position, thus producing a 2-D image in any desired direction. Projection images are used as guides to set up the parameters for a CAT scanner, including tilt, number of slices, and range. Figures 7.3(b) through (h) were obtained by reducing the number of bits from k=7 to k=1 while keeping the spatial resolution constant at 452*374 pixels. The 256-, 128-, and 64-level images are visually identical for all practical purposes. The 32-level image shown in Fig. 7.3 (d), however, has an almost imperceptible set of very fine ridge like structures in areas of smooth gray levels (particularly in the skull).This effect, caused by the use of an insufficient number of gray levels in smooth areas of a digital image, is called false contouring, so called because the ridges resemble topographic contours in a map. False contouring generally is quite visible in images displayed using 16 or less uniformly spaced gray levels, as the images in Figs. 7.3(e) through (h) show.



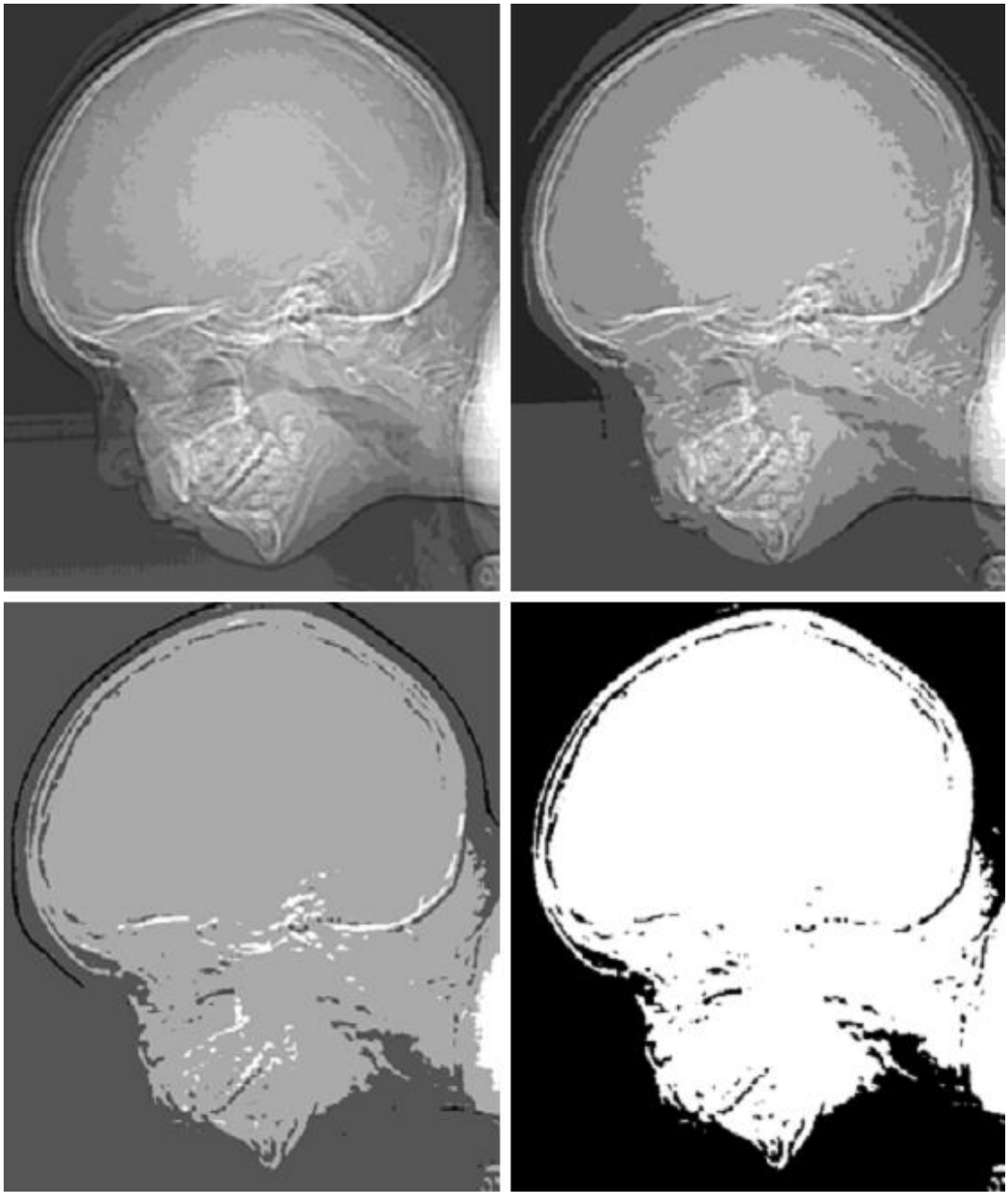


Fig. 7.3 (a) 452*374, 256-level image (b)–(d) Image displayed in 128, 64, and 32 gray levels, while keeping the spatial resolution constant (e)–(g) Image displayed in 16, 8, 4, and 2 gray levels.

As a very rough rule of thumb, and assuming powers of 2 for convenience, images of size 256*256 pixels and 64 gray levels are about the smallest images that can be expected to be reasonably free of objectionable sampling checker-boards and false contouring.

The results in Examples 7.2 and 7.3 illustrate the effects produced on image quality by varying N and k independently. However, these results only partially answer the question of how varying N and k affect images because we have not considered yet any relationships that might exist between these two parameters.

An early study by Huang [1965] attempted to quantify experimentally the effects on image quality produced by varying N and k simultaneously. The experiment consisted of a set of subjective tests. Images similar to those shown in Fig.7.4 were used. The woman's face is representative of an image with relatively little detail; the picture of the cameraman contains an intermediate amount of detail; and the crowd picture contains, by comparison, a large amount of detail. Sets of these three types of images were generated by varying N and k, and observers were then asked to rank them according to their subjective quality. Results were summarized in the form of so-called isopreference curves in the Nk-plane (Fig.7.5 shows average isopreference curves representative of curves corresponding to the images shown in Fig. 7.4).Each point in the Nk-plane represents an image having values of N and k equal to the coordinates of that point.

a b c

Fig.7.4 (a) Image with a low level of detail (b) Image with a medium level of detail (c) Image with a relatively large amount of detail

Points lying on an isopreference curve correspond to images of equal subjective quality. It was found in the course of the experiments that the isopreference curves tended to shift right and upward, but their shapes in each of the three image categories were similar to those shown in

Fig. 7.5. This is not unexpected, since a shift up and right in the curves simply means larger values for N and k, which implies better picture quality.

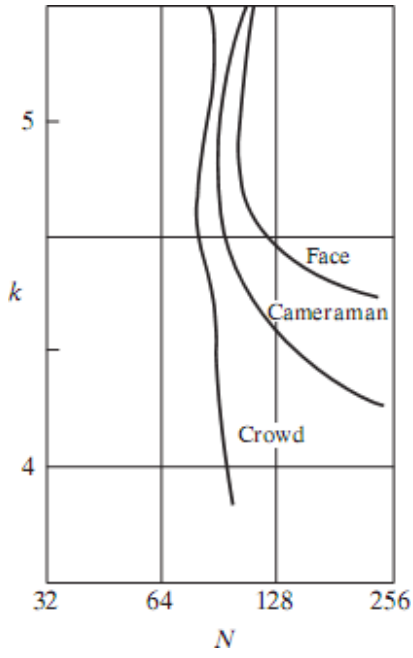


Fig.7.5. Representative isopreference curves for the three types of images in Fig.7.4

The key point of interest in the context of the present discussion is that isopreference curves tend to become more vertical as the detail in the image increases. This result suggests that for images with a large amount of detail only a few gray levels may be needed. For example, the isopreference curve in Fig.7.5 corresponding to the crowd is nearly vertical. This indicates that, for a fixed value of N, the perceived quality for this type of image is nearly independent of the number of gray levels used. It is also of interest to note that perceived quality in the other two image categories remained the same in some intervals in which the spatial resolution was increased, but the number of gray levels actually decreased. The most likely reason for this result is that a decrease in k tends to increase the apparent contrast of an image, a visual effect that humans often perceive as improved quality in an image.

8. Explain about Aliasing and Moire patterns.

Aliasing and Moiré Patterns:

Functions whose area under the curve is finite can be represented in terms of sines and cosines of various frequencies. The sine/cosine component with the highest frequency determines the highest "frequency content" of the function. Suppose that this highest frequency is finite and that the function is of unlimited duration (these functions are called band-limited functions).Then, the Shannon sampling theorem [Brace well (1995)] tells us that, if the function is sampled at a rate equal to or greater than twice its highest frequency, it is possible to recover completely the original function from its samples. If the function is undersampled, then a phenomenon called aliasing corrupts the sampled image. The corruption is in the form of additional frequency components being introduced into the sampled function. These are called aliased frequencies. Note that the sampling rate in images is the number of samples taken (in both spatial directions) per unit distance.

As it turns out, except for a special case discussed in the following paragraph, it is impossible to satisfy the sampling theorem in practice. We can only work with sampled data that are finite in duration. We can model the process of converting a function of unlimited duration into a function of finite duration simply by multiplying the unlimited function by a "gating function" that is valued 1 for some interval and 0 elsewhere. Unfortunately, this function itself has frequency components that extend to infinity. Thus, the very act of limiting the duration of a band-limited function causes it to cease being band limited, which causes it to violate the key condition of the sampling theorem. The principal approach for reducing the aliasing effects on an image is to reduce its high-frequency components by blurring the image prior to sampling. However, aliasing is always present in a sampled image. The effect of aliased frequencies can be seen under the right conditions in the form of so called Moiré patterns.

There is one special case of significant importance in which a function of infinite duration can be sampled over a finite interval without violating the sampling theorem. When a function is periodic, it may be sampled at a rate equal to or exceeding twice its highest frequency and it is possible to recover the function from its samples provided that the sampling captures exactly an integer number of periods of the function. This special case allows us to illustrate vividly the Moiré effect. Figure 8 shows two identical periodic patterns of equally spaced vertical bars, rotated in opposite directions and then superimposed on each other by multiplying the two images. A Moiré pattern, caused by a breakup of the periodicity, is seen in Fig.8 as a 2-D sinusoidal (aliased) waveform (which looks like a corrugated tin roof) running in a vertical direction. A similar pattern can appear when images are digitized (e.g., scanned) from a printed page, which consists of periodic ink dots.

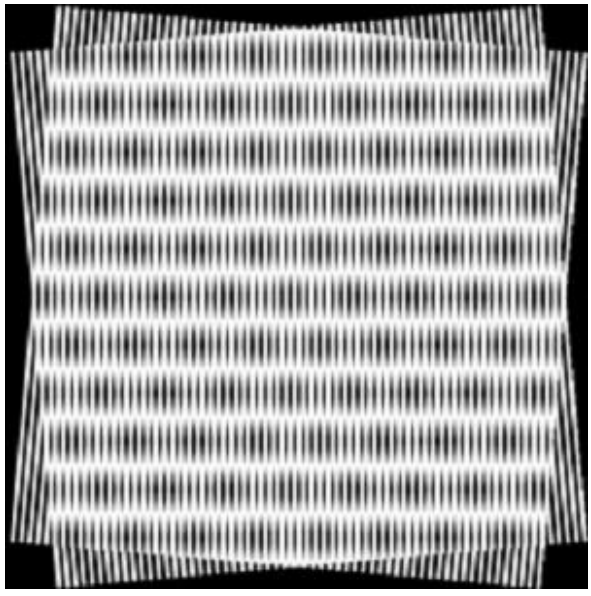


Fig.8. Illustration of the Moiré pattern effect

9. Explain about the basic relationships and distance measures between pixels in a digital image.

Neighbors of a Pixel:

A pixel p at coordinates (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 neighbors 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 (x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1) and are denoted by N_D (p). These points, together with the 4-neighbors, are called the 8-neighbors of p, denoted by N_8 (p). As before, some of the points in N_D (p) and N_8 (p) fall outside the image if (x, y) is on the border of the image.

Connectivity:

Connectivity between pixels is a fundamental concept that simplifies the definition of numerous digital image concepts, such as regions and boundaries. To establish if two pixels are connected, it must be determined if they are neighbors and if their gray levels satisfy a specified criterion of similarity (say, if their gray levels are equal). For instance, in a binary image with values 0 and 1, two pixels may be 4-neighbors, but they are said to be connected only if they have the same value.

Let V be the set of gray-level values used to define adjacency. In a binary image, V={1} if we are referring to adjacency of pixels with value 1. In a grayscale 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 gray-level 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 m-adjacent if

(i) q is in N_4 (p), or

(ii) q is in N_D (p) and the set has no pixels whose values are from V.

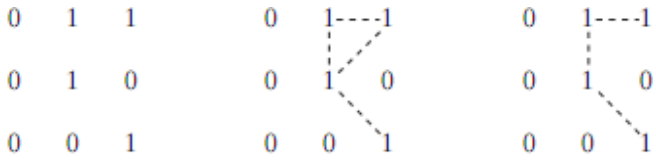
Mixed adjacency is a modification of 8-adjacency. It is introduced to eliminate the ambiguities that often arise when 8-adjacency is used. For example, consider the pixel arrangement shown in Fig.9 (a) for V= {1}.The three pixels at the top of Fig.9 (b) show multiple (ambiguous) 8-adjacency, as indicated by the dashed lines. This ambiguity is removed by using m-adjacency, as shown in Fig. 9 (c).Two image subsets S1 and S2 are adjacent if some pixel in S1 is adjacent to some pixel in S2. It is understood here and in the following definitions that adjacent means 4-, 8-, or m-adjacent. A (digital) path (or curve) from pixel p with coordinates (x, y) to pixel q with coordinates (s, t) is a sequence of distinct pixels with coordinates

$$(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$$

where $(x_0, y_0) = (x, y), (x_n, y_n) = (s, t)$, and pixels (x_i, y_i) and (x_{i-1}, y_{i-1}) are adjacent for $1 \leq i \leq n$. In this case, n is the length of the path. If $(x_o, y_o) = (x_n, y_n)$, the path is a closed path. We can define 4-, 8-, or m-paths depending on the type of adjacency specified. For example, the paths shown in Fig. 9 (b) between the northeast and southeast points are 8-paths, and the path in Fig. 9 (c) is an m-path. Note the absence of ambiguity in the m-path. Let S represent a subset of pixels in an image. Two pixels p and q are said to be connected in S if there

exists a path between them consisting entirely of pixels in S. For any pixel p in S, the set of pixels that are connected to it in S is called a connected component of S. If it only has one connected component, then set S is called a connected set.

Let R be a subset of pixels in an image. We call R a region of the image if R is a connected set. The boundary (also called border or contour) of a region R is the set of pixels in the region that have one or more neighbors that are not in R. If R happens to be an entire image (which we recall is a rectangular set of pixels), then its boundary is defined as the set of pixels in the first and last rows and columns of the image. This extra definition is required because an image has no neighbors beyond its border. Normally, when we refer to a region, we are referring to a subset



a b c

Fig.9 (a) Arrangement of pixels; (b) pixels that are 8-adjacent (shown dashed) to the center pixel; (c) m-adjacency

of an image, and any pixels in the boundary of the region that happen to coincide with the border of the image are included implicitly as part of the region boundary.

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

(a) $D(p, q) \geq 0 \quad (D(p, q) = 0 \quad \text{iff} \quad p = q)$,
(b) $D(p, q) = D(q, p)$, and
(c) $D(p, z) \leq D(p, q) + D(q, z)$.

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

$$D_e(p, q) = \left[(x - s)^2 + (y - t)^2\right]^{\frac{1}{2}}.$$

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 (also called 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:

$$\begin{array}{ccccc} & & 2 & & \\ & 2 & 1 & 2 & \\ 2 & 1 & 0 & 1 & 2 \\ & 2 & 1 & 2 & \\ & & 2 & & \end{array}$$

The pixels with D_4 =1 are the 4-neighbors of (x, y).

The **D_8 distance (also called chessboard distance)** between p and q is defined as

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

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

$$\begin{array}{ccccc} 2 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{array}$$

The pixels with D_8=1 are the 8-neighbors of (x, y). Note that the D4 and D8 distances between p and q are independent of any paths that might exist between the points because these distances involve only the coordinates of the points. If we elect to consider m-adjacency, however, the D_m distance between two points is defined as the shortest m-path between the points. In this case, the distance between two pixels will depend on the values of the pixels along the path, as well as the values of their neighbors. For instance, consider the following arrangement of pixels and assume that p, p_2 , and p_4 have value 1 and that p_1 and p_3 can have a value of 0 or 1:

$$\begin{matrix} & p_3 & p_4 \\ p_1 & p_2 & \\ p & & \end{matrix}$$

Suppose that we consider adjacency of pixels valued 1 (i.e. = {1}). If p_1 and p_3 are 0, the length of the shortest m-path (the Dm distance) between p and p_4 is 2. If p1 is 1, then p_2 and p will no longer be m-adjacent (see the definition of m-adjacency) and the length of the shortest m-path becomes 3 (the path goes through the points $pp_1p_2p_4$). Similar comments apply if p_3 is 1 (and p_1 is 0); in this case, the length of the shortest m-path also is 3. Finally, if both p_1 and p_3 are 1 the length of the shortest m-path between p and p4 is 4. In this case, the path goes through the sequence of points $pp_1p_2p_3p_4$.

10. Write about perspective image transformation.

A perspective transformation (also called an imaging transformation) projects 3D points onto a plane. Perspective transformations play a central role in image processing because they provide an approximation to the manner in which an image is formed by viewing a 3D world. These transformations are fundamentally different, because they are nonlinear in that they involve division by coordinate values.

Figure 10 shows a model of the image formation process. The camera coordinate system (x, y, z) has the image plane coincident with the xy plane and the optical axis (established by the center of the lens) along the z axis. Thus the center of the image plane is at the origin, and the centre of the lens is at coordinates (0.0, λ). If the camera is in focus for distant objects, λ is the focal length of the lens. Here the assumption is that the camera coordinate system is aligned with the world coordinate system (X, Y, Z).

Let (X, Y, Z) be the world coordinates of any point in a 3-D scene, as shown in the Fig. 10. We assume throughout the following discussion that Z> λ; that is all points of interest lie in front of the lens. The first step is to obtain a relationship that gives the coordinates (x, y) of the projection of the point (X. Y, Z) onto the image plane. This is easily accomplished by the use of similar triangles. With reference to Fig. 10,

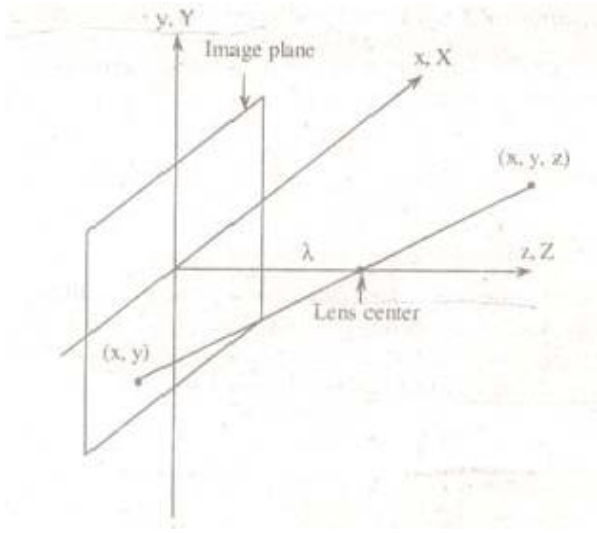


Fig.10 Basic model of the imaging process The camera coordinate system (x, y, z) is aligned with the world coordinate system (X, Y, Z)

$$\frac{x}{\lambda} = -\frac{X}{Z-\lambda} = \frac{X}{\lambda - Z}$$

$$\frac{y}{\lambda} = -\frac{Y}{Z-\lambda} = \frac{Y}{\lambda - Z}$$

Where the negative signs in front of X and Y indicate that image points are actually inverted, as the geometry of Fig.10 shows.
The image-plane coordinates of the projected 3-D point follow directly from above equations

UNIT -II

IMAGE ENHANCEMENT

Image enhancement approaches fall into two broad categories: spatial domain methods and frequency domain methods. The term spatial domain refers to the image plane itself, and approaches in this category are based on direct manipulation of pixels in an image.

Frequency domain processing techniques are based on modifying the Fourier transform of an image. Enhancing an image provides better contrast and a more detailed image as compare to non enhanced image. Image enhancement has very good applications. It is used to enhance medical images, images captured in remote sensing, images from satellite e.t.c. As indicated previously, the term spatial domain refers to the aggregate of pixels composing an image. Spatial domain methods are procedures that operate directly on these pixels. Spatial domain processes will be denoted by the expression.

$$g(x,y) = T[f(x,y)]$$

where f(x, y) is the input image, g(x, y) is the processed image, and T is an operator on f, defined over some neighborhood of (x, y). The principal approach in defining a neighborhood about a point (x, y) is to use a square or rectangular subimage area centered at (x, y), as Fig. 2.1 shows. The center of the subimage is moved from pixel to pixel starting, say, at the top left corner. The operator T is applied at each location (x, y) to yield the output, g, at that location. The process utilizes only the pixels in the area of the image spanned by the neighborhood.

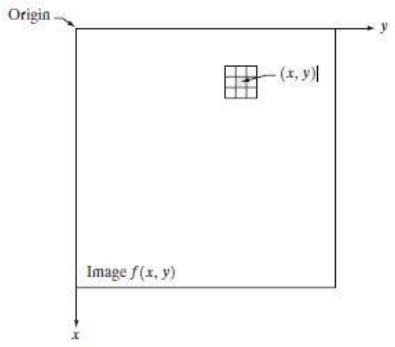


Fig.: 3x3 neighborhood about a point (x,y) in an image.

The simplest form of T is when the neighborhood is of size 1*1 (that is, a single pixel). In this case, g depends only on the value of f at (x, y), and T becomes a gray-level (also called an intensity or mapping) transformation function of the form

$$s = T (r)$$

where r is the pixels of the input image and s is the pixels of the output image. T is a transformation function that maps each value of 'r' to each value of 's'.

For example, if T(r) has the form shown in Fig. 2.2(a), the effect of this transformation would be to produce an image of higher contrast than the original by darkening the levels below m and brightening the levels above m in the original image. In this technique, known as contrast stretching, the values of r below m are compressed by the transformation function into a narrow range of s, toward black.The opposite effect takes place for values of r above m.

In the limiting case shown in Fig. 2.2(b), T(r) produces a two-level (binary) image. A mapping of this form is called a thresholding function.

One of the principal approaches in this formulation is based on the use of so-called masks (also referred to as filters, kernels, templates, or windows). Basically, a mask is a small (say, 3*3) 2-D array, such as the one shown in Fig. 2.1, in which the values of the mask coefficients determine the nature of the process, such as image sharpening. Enhancement techniques based on this type of approach often are referred to as mask processing or filtering.

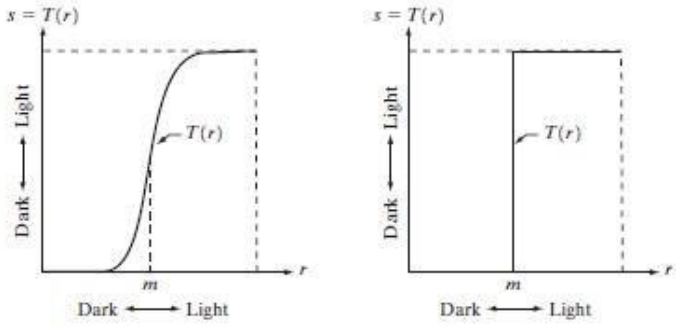


Fig. 2.2 Gray level transformation functions for contrast enhancement.

Image enhancement can be done through gray level transformations which are discussed below.

BASIC GRAY LEVEL TRANSFORMATIONS:

- Image negative
- Log transformations
- Power law transformations
- Piecewise-Linear transformation functions

LINEAR TRANSFORMATION:

First we will look at the linear transformation. Linear transformation includes simple identity and negative transformation. Identity transformation has been discussed in our tutorial of image transformation, but a brief description of this transformation has been given here.

Identity transition is shown by a straight line. In this transition, each value of the input image is directly mapped to each other value of output image. That results in the same input image and output image. And hence is called identity transformation. It has been shown below:

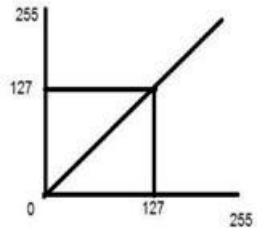


Fig. Linear transformation between input and output.

NEGATIVE TRANSFORMATION:

The second linear transformation is negative transformation, which is invert of identity transformation. In negative transformation, each value of the input image is subtracted from the L-1 and mapped onto the output image

IMAGE NEGATIVE: The image negative with gray level value in the range of [0, L-1] is obtained by negative transformation given by S = T(r) or

$$S = L - 1 - r$$

Where r= gray level value at pixel (x,y)

L is the largest gray level consists in the image

It results in getting photograph negative. It is useful when for enhancing white details embedded in dark regions of the image.

The overall graph of these transitions has been shown below.

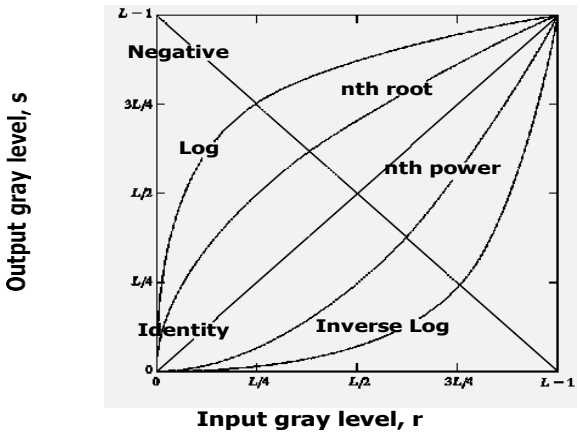


Fig. Some basic gray-level transformation functions used for image enhancement.

In this case the following transition has been done.

$$S = (L - 1) - r$$

since the input image of Einstein is an 8 bpp image, so the number of levels in this image are 256. Putting 256 in the equation, we get this

$$S = 255 - r$$

So each value is subtracted by 255 and the result image has been shown above. So what happens is that, the lighter pixels become dark and the darker picture becomes light. And it results in image negative.

It has been shown in the graph below.

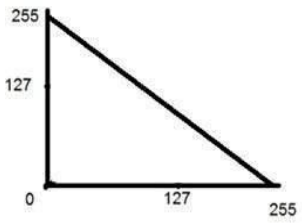


Fig. Negative transformations.

LOGARITHMIC TRANSFORMATIONS:

Logarithmic transformation further contains two type of transformation. Log transformation and inverse log transformation.

LOG TRANSFORMATIONS:

The log transformations can be defined by this formula

$$s = c \log(r + 1).$$

Where s and r are the pixel values of the output and the input image and c is a constant. The value 1 is added to each of the pixel value of the input image because if there is a pixel intensity of 0 in the image, then log (0) is equal to infinity. So 1 is added, to make the minimum value at least 1.

During log transformation, the dark pixels in an image are expanded as compare to the higher pixel values. The higher pixel values are kind of compressed in log transformation. This result in following image enhancement.

An another way of representing LOG TRANSFORMATIONS: Enhance details in the darker regions of an image at the expense of detail in brighter regions.

$$T(f) = C * \log (1+r)$$

- Here C is constant and $r \geq 0$.
- The shape of the curve shows that this transformation maps the narrow range of low gray level values in the input image into a wider range of output image.
- The opposite is true for high level values of input image.

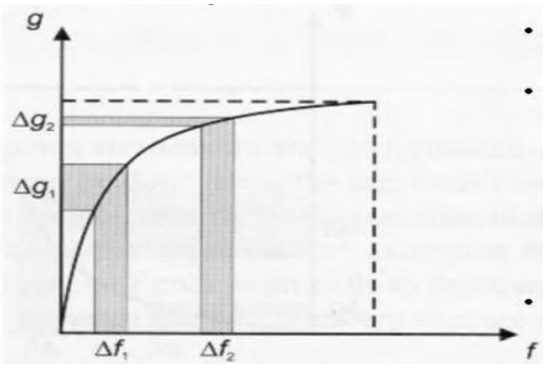


Fig. log transformation curve input vs output

POWER – LAW TRANSFORMATIONS:

There are further two transformation is power law transformations, that include nth power and nth root transformation. These transformations can be given by the expression:

$$s=cr^{\gamma}$$

This symbol γ is called gamma, due to which this transformation is also known as gamma transformation.

Variation in the value of γ varies the enhancement of the images. Different display devices / monitors have their own gamma correction, that's why they display their image at different intensity.

where c and g are positive constants. Sometimes Eq. (6) is written as $S = C (r + \varepsilon)^{\gamma}$ to account for an offset (that is, a measurable output when the input is zero). Plots of s versus r for various values of γ are shown in Fig. 2.10. As in the case of the log transformation, power-law curves with fractional values of γ map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher values of input levels. Unlike the log function, however, we notice here a family of possible transformation curves obtained simply by varying γ.

In Fig that curves generated with values of γ>1 have exactly The opposite effect as those generated with values of γ<1. Finally, we Note that Eq. (6) reduces to the identity transformation when c=γ=1.

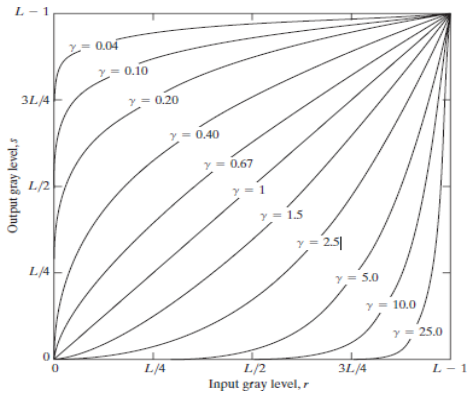


Fig. 2.13 Plot of the equation $S = cr^{\gamma}$ for various values of γ (c =1 in all cases).

This type of transformation is used for enhancing images for different type of display devices. The gamma of different display devices is different. For example Gamma of CRT lies in between of 1.8 to 2.5, that means the image displayed on CRT is dark.

Varying gamma (γ) obtains family of possible transformation curves $S = C^* r^{\gamma}$

Here C and γ are positive constants. Plot of S versus r for various values of γ is

γ > 1 compresses dark values

Expands bright values

γ < 1 (similar to Log transformation)

Expands dark values

Compresses bright values

When C = γ = 1 , it reduces to identity transformation .

CORRECTING GAMMA:

$s=cr^{\gamma}$

$s=cr^{(1/2.5)}$

The same image but with different gamma values has been shown here.

Piecewise-Linear Transformation Functions:

A complementary approach to the methods discussed in the previous three sections is to use piecewise linear functions. The principal advantage of piecewise linear functions over the types of functions which we have discussed thus far is that the form of piecewise functions can be arbitrarily complex.

The principal disadvantage of piecewise functions is that their specification requires considerably more user input.

Contrast stretching: One of the simplest piecewise linear functions is a contrast-stretching transformation. Low-contrast images can result from poor illumination, lack of dynamic range in the imaging sensor, or even wrong setting of a lens aperture during image acquisition.

$$S= T(r)$$

Figure x(a) shows a typical transformation used for contrast stretching. The locations of points (r1, s1) and (r2, s2) control the shape of the transformation

Function. If r1=s1 and r2=s2, the transformation is a linear function that produces No changes in gray levels. If r1=r2, s1=0and s2= L-1, the transformation Becomes a thresholding function that creates a binary image, as illustrated In fig. 2.2(b).

Intermediate values of ar1, s1b and ar2, s2b produce various degrees Of spread in the gray levels of the output image, thus affecting its contrast. In general, $r1 \leq r2$ and $s1 \leq s2$ is assumed so that the function is single valued and Monotonically increasing.

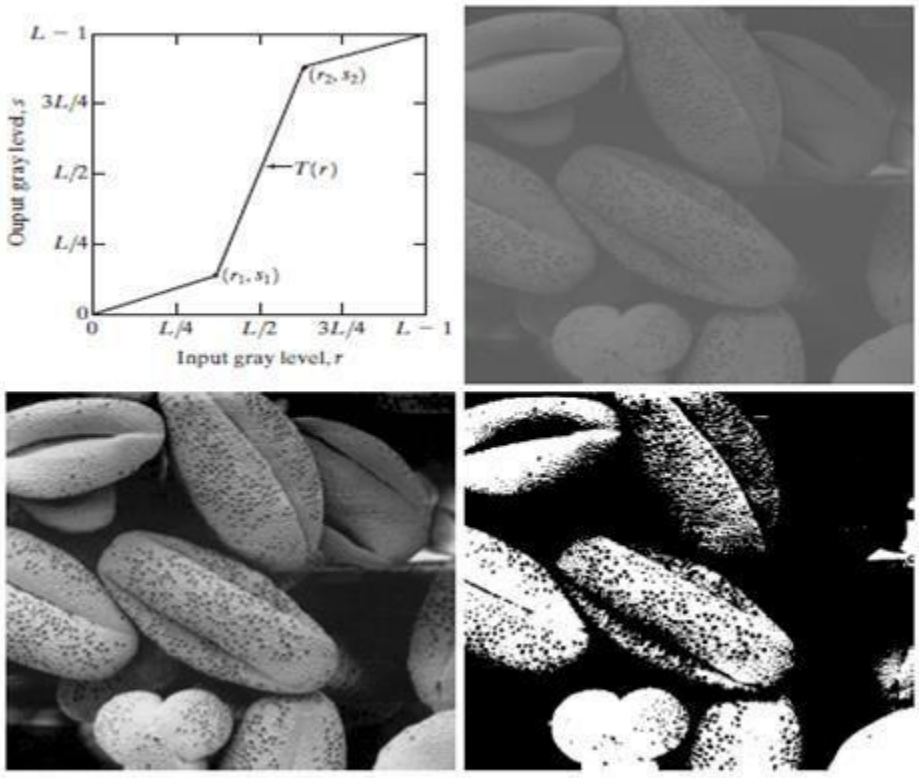


Fig. x Contrast stretching. (a) Form of transformation function. (b) A low-contrast stretching. (c) Result of high contrast stretching. (d) Result of thresholding (original image courtesy of Dr.Roger Heady, Research School of Biological Sciences, Australian National University Canberra Australia.

Figure x(b) shows an 8-bit image with low contrast. Fig. x(c) shows the result of contrast stretching, obtained by setting (r1, s1)=(rmin, 0) and (r2, s2)=(r_{max},L-1) where r_{min} and r_{max} denote the minimum and maximum gray levels in the image, respectively.Thus, the transformation function stretched the levels linearly from their original range to the full range [0, L-1]. Finally, Fig. x(d) shows the result of using the thresholding function defined previously, with r1=r2=m, the mean gray level in the image. The original image on which these results are based is a scanning electron microscope image of pollen, magnified approximately 700 times.

Gray-level slicing:

Highlighting a specific range of gray levels in an image often is desired. Applications include enhancing features such as masses of water in satellite imagery and enhancing flaws in X-ray images.

There are several ways of doing level slicing, but most of them are variations of two basic themes. One approach is to display a high value for all gray levels in the range of interest and a low value for all other gray levels.

This transformation, shown in Fig. y(a), produces a binary image. The second approach, based on the transformation shown in Fig.y (b), brightens the desired range of gray levels but preserves the background and gray-level tonalities in the image. Figure y (c) shows a gray-scale image, and Fig. y(d) shows the result of using the transformation in Fig. y(a).Variations of the two transformations shown in Fig. are easy to formulate.

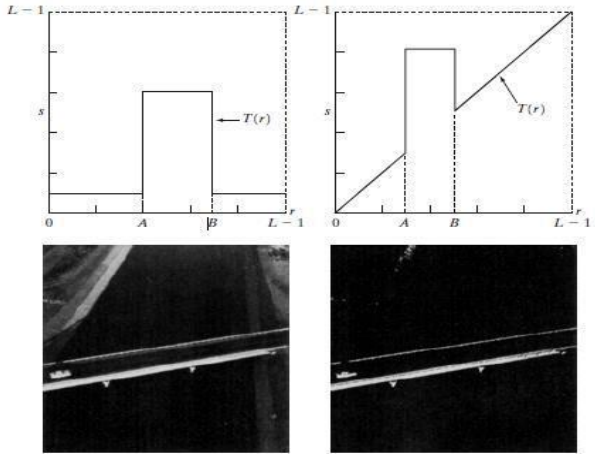


Fig. y (a)This transformation highlights range [A,B] of gray levels and reduces all others to a constant level (b) This transformation highlights range [A,B] but preserves all other levels. (c) An image . (d) Result of using the transformation in (a).

BIT-PLANE SLICING:

Instead of highlighting gray-level ranges, highlighting the contribution made to total image appearance by specific bits might be desired. Suppose that each pixel in an image is represented by 8 bits. Imagine that the image is composed of eight 1-bit planes, ranging from bit-plane 0 for the least significant bit to bit plane 7 for the most significant bit. In terms of 8-bit bytes, plane 0 contains all the lowest order bits in the bytes comprising the pixels in the image and plane 7 contains all the high-order bits.

Figure 3.12 illustrates these ideas, and Fig. 3.14 shows the various bit planes for the image shown in Fig. 3.13. Note that the higher-order bits (especially the top four) contain the majority of the visually significant data. The other bit planes contribute to more subtle details in the image. Separating a digital image into its bit planes is useful for analyzing the relative

importance played by each bit of the image, a process that aids in determining the adequacy of the number of bits used to quantize each pixel.

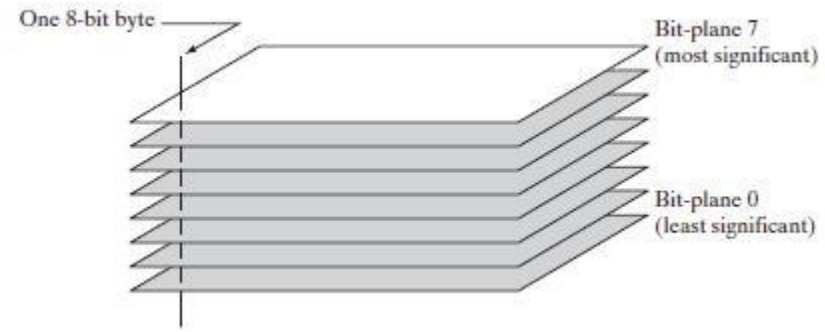


FIGURE Bit-plane representation of an 8-bit image.

In terms of bit-plane extraction for an 8-bit image, it is not difficult to show that the (binary) image for bit-plane 7 can be obtained by processing the input image with a thresholding gray-level transformation function that (1) maps all levels in the image between 0 and 127 to one level (for example, 0); and (2) maps all levels between 129 and 255 to another (for example, 255).The binary image for bit-plane 7 in Fig. 3.14 was obtained in just this manner. It is left as an exercise

(Problem 3.3) to obtain the gray-level transformation functions that would yield the other bit planes.

Histogram Processing:

The histogram of a digital image with gray levels in the range [0, L-1] is a discrete function of the form

$$H(r_k)=n_k$$

where r_k is the k^{th} gray level and n_k is the number of pixels in the image having the level r_k..

A normalized histogram is given by the equation

$$p(r_k)=n_k/n \text{ for } k=0,1,2,\ldots..,L-1$$

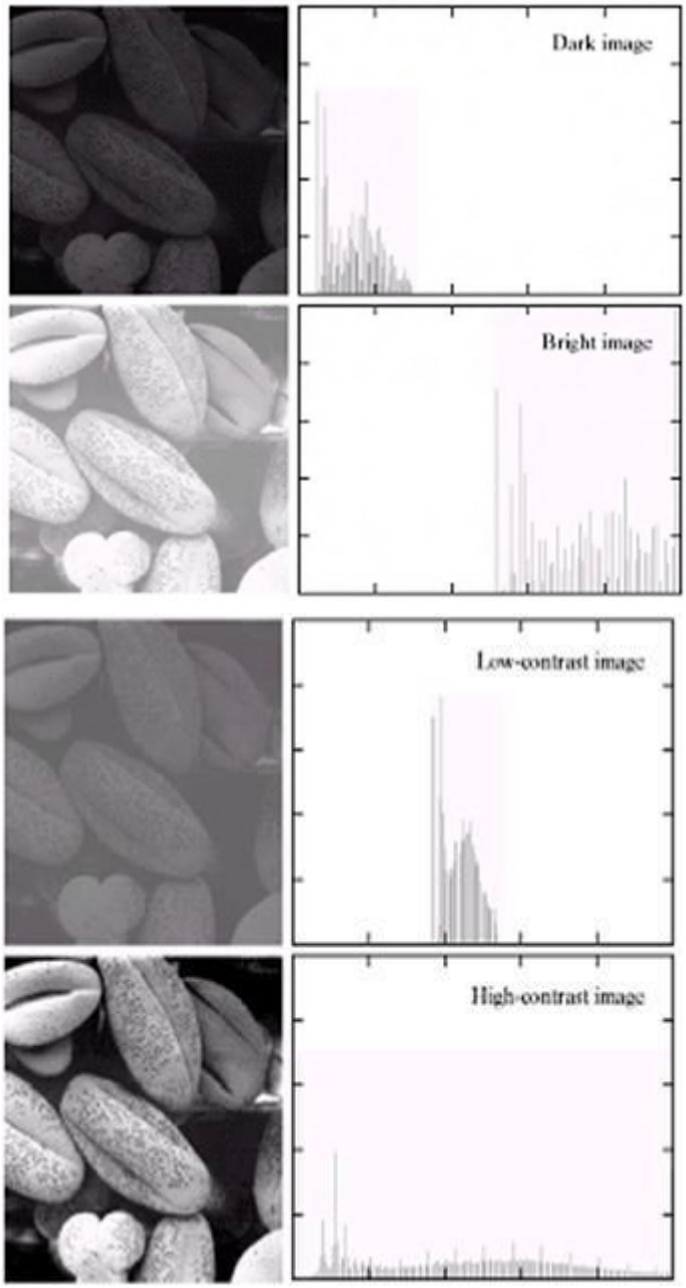
$P(r_k)$ gives the estimate of the probability of occurrence of gray level r_k.

The sum of all components of a normalized histogram is equal to 1.

The histogram plots are simple plots of $p(r_k)=n_k$ versus r_k.

In the dark image the components of the histogram are concentrated on the low (dark) side of the gray scale. In case of bright image the histogram components are baised towards the high side of the gray scale. The histogram of a low contrast image will be narrow and will be centered towards the middle of the gray scale.

The components of the histogram in the high contrast image cover a broad range of the gray scale. The net effect of this will be an image that shows a great deal of gray levels details and has high dynamic range.



Histogram Equalization:

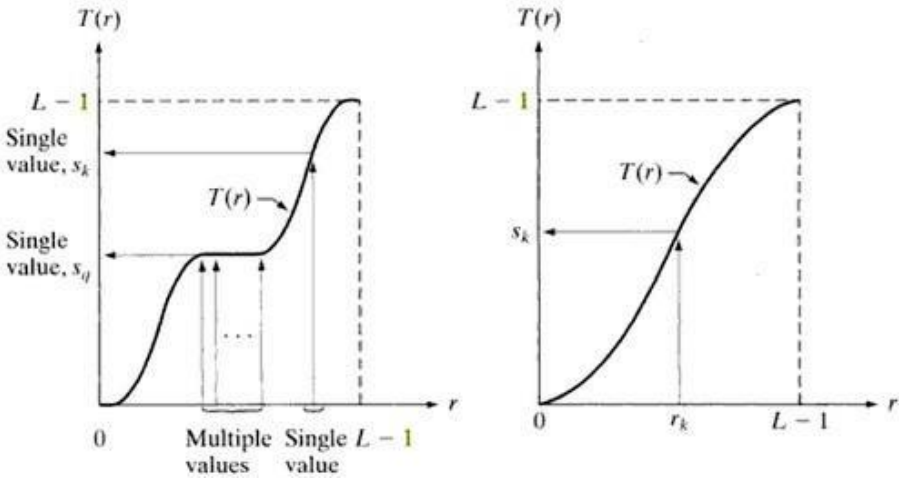
Histogram equalization is a common technique for enhancing the appearance of images. Suppose we have an image which is predominantly dark. Then its histogram would be skewed towards the lower end of the grey scale and all the image detail are compressed into the dark

end of the histogram. If we could 'stretch out' the grey levels at the dark end to produce a more uniformly distributed histogram then the image would become much clearer.

Let there be a continuous function with r being gray levels of the image to be enhanced. The range of r is [0, 1] with r=0 representing black and r=1 representing white. The transformation function is of the form

$$S=T(r) \text{ where } 0<r<1$$

It produces a level s for every pixel value r in the original image.



a b

FIGURE (a) Monotonically increasing function, showing how multiple values can map to a single value. (b) Strictly monotonically increasing function. This is a one-to-one mapping, both ways.

The transformation function is assumed to fulfill two condition T(r) is single valued and monotonically increasing in the internal 0<T(r)<1 for 0<r<1.The transformation function should be single valued so that the inverse transformations should exist. Monotonically increasing condition preserves the increasing order from black to white in the output image. The second conditions guarantee that the output gray levels will be in the same range as the input levels. The gray levels of the image may be viewed as random variables in the interval [0.1]. The most fundamental descriptor of a random variable is its probability density function (PDF) Pr(r) and Ps(s) denote the probability density functions of random variables r and s respectively. Basic results from an elementary probability theory states that if Pr(r) and Tr are known and T-1(s) satisfies conditions (a), then the probability density function Ps(s) of the transformed variable is given by the formula

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$

Thus the PDF of the transformed variable s is the determined by the gray levels PDF of the input image and by the chosen transformations function.

A transformation function of a particular importance in image processing

$$s = T(r) = (L - 1)\int_0^r p_r(w)\,dw$$

This is the cumulative distribution function of r.

L is the total number of possible gray levels in the image.

IMAGE ENHANCEMENT IN FREQUENCY DOMAIN

BLURRING/NOISE REDUCTION: Noise characterized by sharp transitions in image intensity. Such transitions contribute significantly to high frequency components of Fourier transform. Intuitively, attenuating certain high frequency components result in blurring and reduction of image noise.

IDEAL LOW-PASS FILTER:

Cuts off all high-frequency components at a distance greater than a certain distance from origin (cutoff frequency).

$$H(u,v) = 1, \text{ if } D(u,v) \le D_0$$

$$0, \text{ if } D(u,v) > D_0$$

Where D0 is a positive constant and D(u,v) is the distance between a point (u,v) in the frequency domain and the center of the frequency rectangle; that is

$$D(u,v) = [(u-P/2)^2 + (v-Q/2)^2]^{1/2}$$

Where as P and Q are the padded sizes from the basic equations

Wraparound error in their circular convolution can be avoided by padding these functions with zeros,

VISUALIZATION: IDEAL LOW PASS FILTER:

Aa shown in fig.below

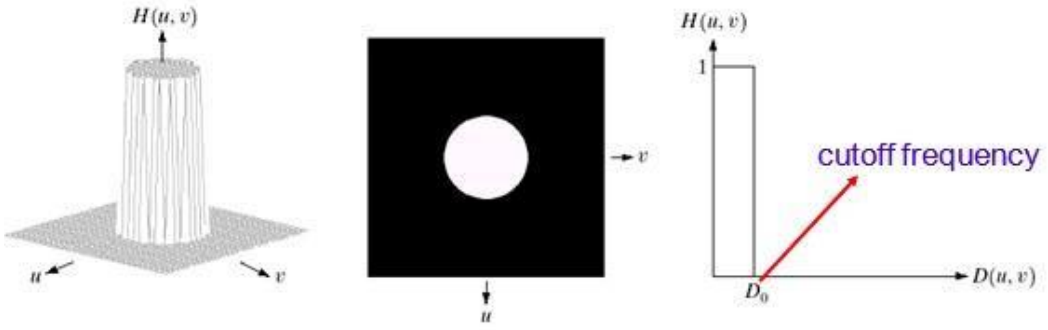


Fig: ideal low pass filter 3-D view and 2-D view and line graph.

EFFECT OF DIFFERENT CUT OFF FREQUENCIES:

Fig.below(a) Test pattern of size 688x688 pixels, and (b) its Fourier spectrum. The spectrum is double the image size due to padding but is shown in half size so that it fits in the page. The superimposed circles have radii equal to 10, 30, 60, 160 and 460 with respect to the full-size spectrum image. These radii enclose 87.0, 93.1, 95.7, 97.8 and 99.2% of the padded image power respectively.

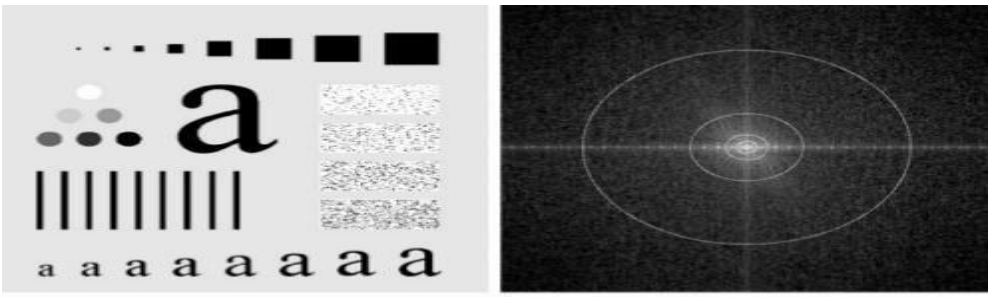


Fig: (a) Test patter of size 688x688 pixels (b) its Fourier spectrum

Fig: (a) original image, (b)-(f) Results of filtering using ILPFs with cutoff frequencies set at radii values 10, 30, 60, 160 and 460, as shown in fig.2.2.2(b). The power removed by these filters was 13, 6.9, 4.3, 2.2 and 0.8% of the total, respectively.

As the cutoff frequency decreases,

- image becomes more blurred
- Noise becomes increases
- Analogous to larger spatial filter sizes

The severe blurring in this image is a clear indication that most of the sharp detail information in the picture is contained in the 13% power removed by the filter. As the filter radius is increases less and less power is removed, resulting in less blurring. Fig. (c) through (e) are characterized by “ringing” , which becomes finer in texture as the amount of high frequency content removed decreases.

WHY IS THERE RINGING?

Ideal low-pass filter function is a rectangular function

The inverse Fourier transform of a rectangular function is a sinc function.

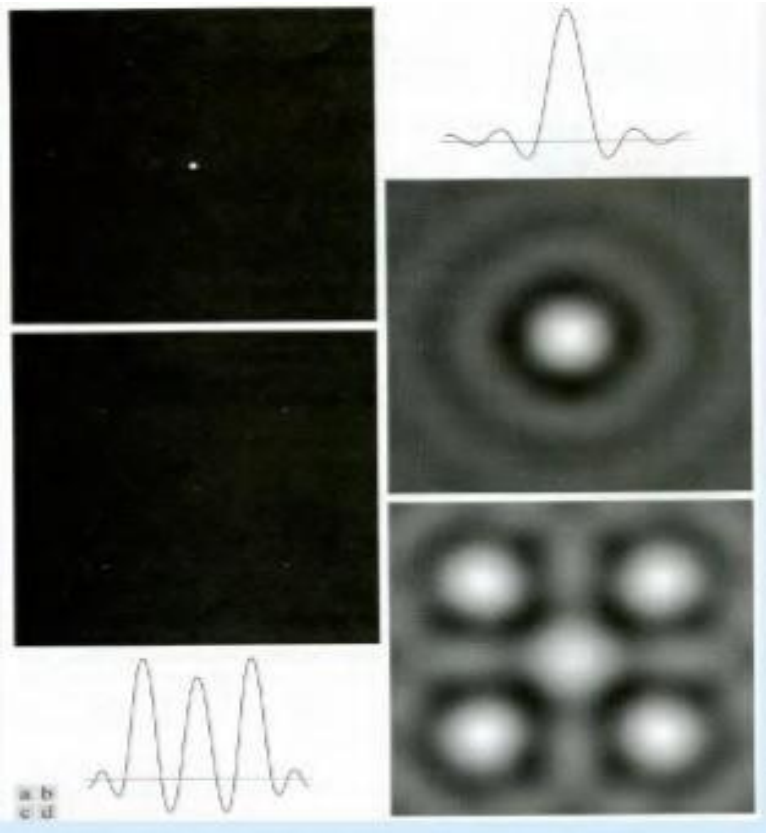


Fig. Spatial representation of ILPFs of order 1 and 20 and corresponding intensity profiles through the center of the filters(the size of all cases is 1000x1000 and the cutoff frequency is 5), observe how ringing increases as a function of filter order.

BUTTERWORTH LOW-PASS FILTER:

Transfor funtion of a Butterworth lowpass filter (BLPF) of order n, and with cutoff frequency at a distance D_0 from the origin, is defined as

- $$H(u,v) = \frac{1}{1+\left[D(u,v)/D_0 \right]^{2n}}$$

Transfer function does not have sharp discontinuity establishing cutoff between passed and filtered frequencies.

Cut off frequency D_0 defines point at which H(u,v) = 0.5

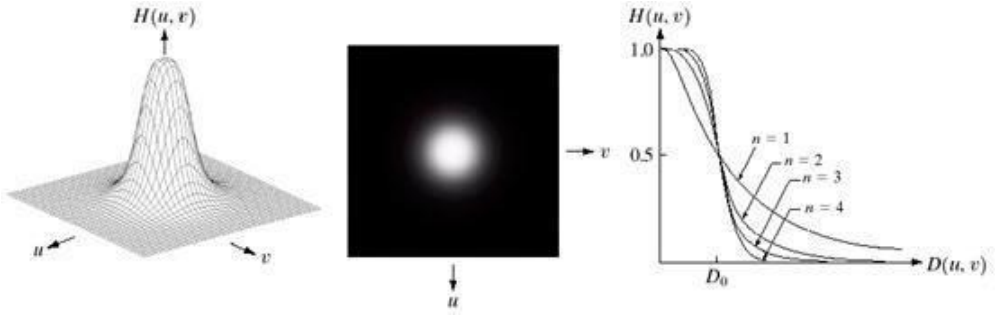


Fig. (a) perspective plot of a Butterworth lowpass-filter transfer function. (b) Filter displayed as an image. (c)Filter radial cross sections of order 1 through 4.

Unlike the ILPF, the BLPF transfer function does not have a sharp discontinuity that gives a clear cutoff between passed and filtered frequencies.

BUTTERWORTH LOW-PASS FILTERS OF DIFFEREN T FREQUENCIES:

Fig. (a) Original image.(b)-(f) Results of filtering using BLPFs of order 2, with cutoff frequencies at the radii

Fig. shows the results of applying the BLPF of eq. to fig.(a), with n=2 and D0 equal to the five radii in fig.(b) for the ILPF, we note here a smooth transition in blurring as a function of increasing cutoff frequency. Moreover, no ringing is visible in any of the images processed with this particular BLPF, a fact attributed to the filter's smooth transition between low and high frequencies.

A BLPF of order 1 has no ringing in the spatial domain. Ringing generally is imperceptible in filters of order 2, but can become significant in filters of higher order.

Fig.shows a comparison between the spatial representation of BLPFs of various orders (using a cutoff frequency of 5 in all cases). Shown also is the intensity profile along a horizontal scan line through the center of each filter. The filter of order 2 does show mild ringing and small negative values, but they certainly are less pronounced than in the ILPF. A butter worth filter of order 20 exhibits characteristics similar to those of the ILPF (in the limit, both filters are identical).

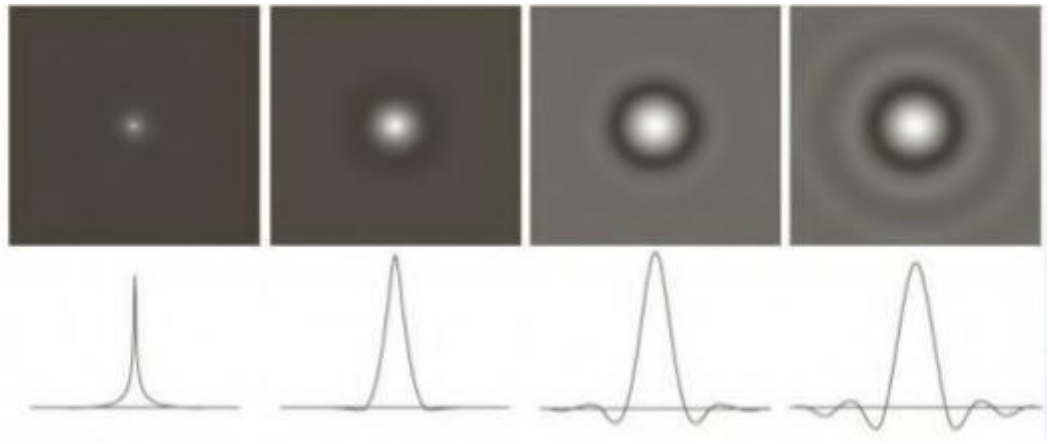


Fig.2.2.7 (a)-(d) Spatial representation of BLPFs of order 1, 2, 5 and 20 and corresponding intensity profiles through the center of the filters (the size in all cases is 1000 x 1000 and the cutoff frequency is 5) Observe how ringing increases as a function of filter order.

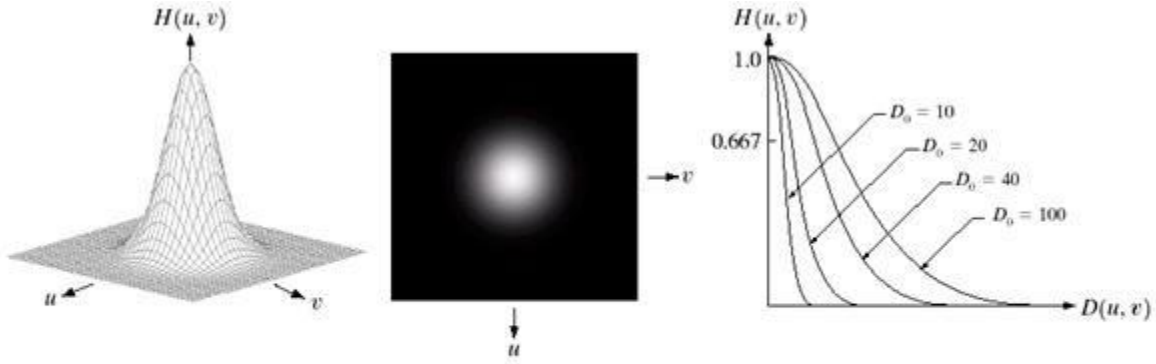
GAUSSIAN LOWPASS FILTERS:

The form of these filters in two dimensions is given by

$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$

- This transfer function is smooth , like Butterworth filter.
- Gaussian in frequency domain remains a Gaussian in spatial domain
- Advantage: No ringing artifacts.

Where D_0 is the cutoff frequency. When D(u,v) = D_0, the GLPF is down to 0.607 of its maximum value. This means that a spatial Gaussian filter, obtained by computing the IDFT of above equation., will have no ringing. Fig..shows a perspective plot, image display and radial cross sections of a GLPF function.



a b c

FIGURE (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of D_0.

Fig. (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c). Filter radial cross sections for various values of D_0

Fig.(a) Original image. (b)-(f) Results of filtering using GLPFs with cutoff frequencies at the radii shown in fig.2.2.2. compare with fig.2.2.3 and fig.2.2.6

a b c

Fig. (a) Original image (784x 732 pixels). (b) Result of filtering using a GLPF with D0 = 100. (c) Result of filtering using a GLPF with D0 = 80. Note the reduction in fine skin lines in the magnified sections in (b) and (c).

Fig. shows an application of lowpass filtering for producing a smoother, softer-looking result from a sharp original. For human faces, the typical objective is to reduce the sharpness of fine skin lines and small blemished.

IMAGE SHARPENING USING FREQUENCY DOMAIN FILTERS:

An image can be smoothed by attenuating the high-frequency components of its Fourier transform. Because edges and other abrupt changes in intensities are associated with high-frequency components, image sharpening can be achieved in the frequency domain by high pass filtering, which attenuates the low-frequency components without disturbing high-frequency information in the Fourier transform.

The filter function H(u,v) are understood to be discrete functions of size PxQ; that is the discrete frequency variables are in the range u = 0,1,2,…….P-1 and v = 0,1,2,…….Q-1.

The meaning of sharpening is

- Edges and fine detail characterized by sharp transitions in image intensity
- Such transitions contribute significantly to high frequency components of Fourier transform

- Intuitively, attenuating certain low frequency components and preserving high frequency components result in sharpening.

Intended goal is to do the reverse operation of low-pass filters

- When low-pass filter attenuated frequencies, high-pass filter passes them
- When high-pass filter attenuates frequencies, low-pass filter passes them.

A high pass filter is obtained from a given low pass filter using the equation.

$$H_{hp}(u,v) = 1 - H_{tp}(u,v)$$

Where $H_{lp}(u,v)$ is the transfer function of the low-pass filter. That is when the low-pass filter attenuates frequencies, the high-pass filter passed them, and vice-versa.

We consider ideal, Butter-worth, and Gaussian high-pass filters. As in the previous section, we illustrate the characteristics of these filters in both the frequency and spatial domains. Fig.. shows typical 3-D plots, image representations and cross sections for these filters. As before, we see that the Butter-worth filter represents a transition between the sharpness of the ideal filter and the broad smoothness of the Gaussian filter. Fig.discussed in the sections the follow, illustrates what these filters look like in the spatial domain. The spatial filters were obtained and displayed by using the procedure used.

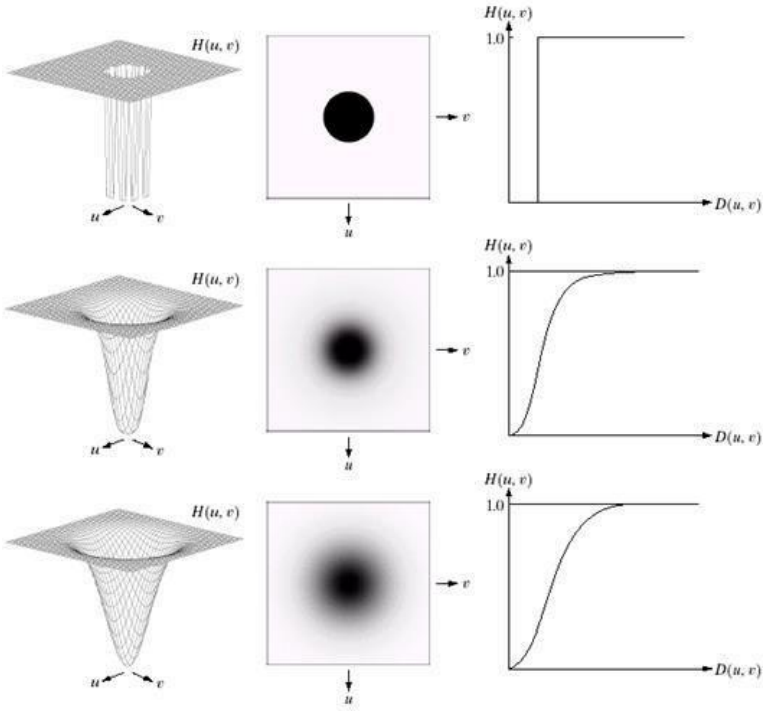


Fig: Top row: Perspective plot, image representation, and cross section of a typical ideal high-pass filter. Middle and bottom rows: The same sequence for typical butter-worth and Gaussian high-pass filters.

IDEAL HIGH-PASS FILTER:

A 2-D ideal high-pass filter (IHPF) is defined as

$$H(u,v) = \begin{cases} 0, & \text{if } D(u,v) \le D_0 \\ 1, & \text{if } D(u,v) > D_0 \end{cases}$$

Where D_0 is the cutoff frequency and D(u,v) is given by eq. As intended, the IHPF is the opposite of the ILPF in the sense that it sets to zero all frequencies inside a circle of radius D_0 while passing, without attenuation, all frequencies outside the circle. As in case of the ILPF, the IHPF is not physically realizable.

SPATIAL REPRESENTATION OF HIGHPASS FILTERS:

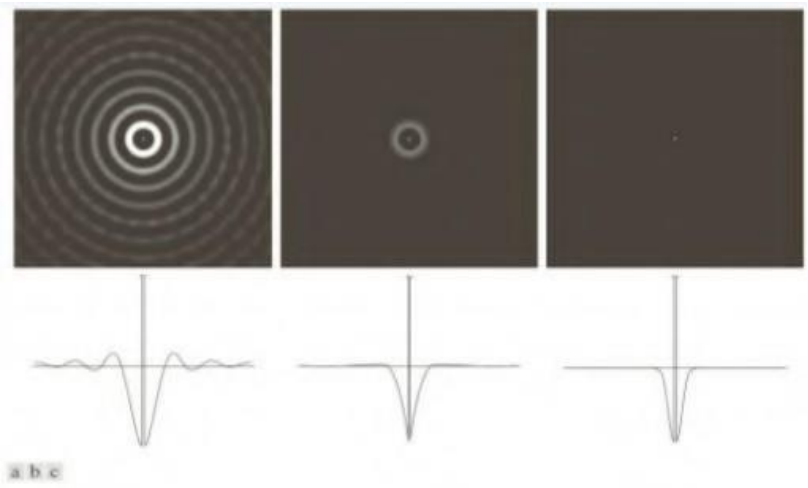


Fig.. Spatial representation of typical (a) ideal (b) Butter-worth and (c) Gaussian frequency domain high-pass filters, and corresponding intensity profiles through their centers.

We can expect IHPFs to have the same ringing properties as ILPFs. This is demonstrated clearly in Fig.. which consists of various IHPF results using the original image in Fig.(a) with D_0 set to 30, 60,and 160 pixels, respectively. The ringing in Fig. (a) is so severe that it produced distorted, thickened object boundaries (e.g.,look at the large letter “a”). Edges of the top three circles do not show well because they are not as strong as the other edges in the image (the intensity of these three objects is much closer to the background intensity, giving discontinuities of smaller magnitude).

FILTERED RESULTS: IHPF:

a b c

Fig.. Results of high-pass filtering the image in Fig.(a) using an IHPF with D_0 = 30, 60, and 160.

The situation improved somewhat with D0 = 60. Edge distortion is quite evident still, but now we begin to see filtering on the smaller objects. Due to the now familiar inverse relationship between the frequency and spatial domains, we know that the spot size of this filter is smaller than the spot of the filter with D0 = 30. The result for D0 = 160 is closer to what a high-pass filtered image should look like. Here, the edges are much cleaner and less distorted, and the smaller objects have been filtered properly.

Of course, the constant background in all images is zero in these high-pass filtered images because highpass filtering is analogous to differentiation in the spatial domain.

BUTTER-WORTH HIGH-PASS FILTERS:

A 2-D Butter-worth high-pass filter (BHPF) of order n and cutoff frequency D_0 is defined as

$$H(u,v) = \frac{1}{1+\left[D_0 / D(u,v)\right]^{2n}}$$

Where D(u,v) is given by Eq.(3). This expression follows directly from Eqs.(3) and (6). The middle row of Fig.2.2.11. shows an image and cross section of the BHPF function.

Butter-worth high-pass filter to behave smoother than IHPFs. Fig.2.2.14.shows the performance of a BHPF of order 2 and with D0 set to the same values as in Fig.2.2.13. The boundaries are much less distorted than in Fig.2.2.13. even for the smallest value of cutoff frequency.

FILTERED RESULTS: BHPF:

Homomorphic Filtering

Homomorphic filters are widely used in image processing for compensating the effect of no uniform illumination in an image. Pixel intensities in an image represent the light reflected from the corresponding points in the objects. As per as image model, image f(z,y) may be characterized by two components: (1) the amount of source light incident on the scene being viewed, and (2) the amount of light reflected by the objects in the scene. These portions of light are called the illumination and reflectance components, and are denoted i (x , y) and r (x , y) respectively. The functions i (x , y) and r (x , y) combine multiplicatively to give the image function f (x , y):

$$f(x, y) = i(x, y).r(x, y) \quad (1)$$

where $0 < i(x, y) < a$ and $0 < r(x, y) < 1$. Homomorphic filters are used in such situations where the image is subjected to the multiplicative interference or noise as depicted in Eq. 1. We cannot easily use the above product to operate separately on the frequency components of illumination and reflection because the Fourier transform of f (x , y) is not separable; that is

$$F[f(x,y)) \text{ not equal to } F[i(x, y)].F[r(x, y)].$$

We can separate the two components by taking the logarithm of the two sides

$$\ln f(x,y) = \ln i(x, y) + \ln r(x, y).$$

Taking Fourier transforms on both sides we get,

$$F[\ln f(x,y)\} = F[\ln i(x, y)\} + F[\ln r(x, y)].$$

that is, F(x,y) = I(x,y) + R(x,y), where F, I and R are the Fourier transforms ln f(x,y),ln i(x, y) , and ln r(x, y). respectively. The function F represents the Fourier transform of the sum of two images: a low-frequency illumination image and a high-frequency reflectance image. If we now apply a filter with a transfer function that suppresses low- frequency components and enhances high-frequency components, then we can suppress the illumination component and enhance the reflectance component. Taking the inverse transform of F (x , y) and then anti-logarithm, we get

fô (x , y) = iô

(x , y) + rô(x, y)

IMAGE RESTORATION

IMAGE RESTORATION

Restoration improves image in some predefined sense. It is an objective process. Restoration attempts to reconstruct an image that has been degraded by using a priori knowledge of the degradation phenomenon. These techniques are oriented toward modeling the degradation and then applying the inverse process in order to recover the original image.

Image Restoration refers to a class of methods that aim to remove or reduce the degradations that have occurred while the digital image was being obtained.

All natural images when displayed have gone through some sort of degradation:

a)During display mode
b)Acquisition mode.
c)Processing mode.

The degradations may be due to

a)Sensor noise
bBlur due to camera mis focus
c)Relative object-camera motion
d)Random atmospheric turbulence
e)Others

A Model of Image Restoration Process

Degradation process operates on a degradation function that operates on an input image with an additive noise term.

Input image is represented by using the notation f(x,y), noise term can be represented as ɖ(x,y).These two terms when combined gives the result as g(x,y).

If we are given g(x,y), some knowledge about the degradation function H or J and some knowledge about the additive noise teem ɖ(x,y), the objective of restoration is to obtain an estimate *f'(x,y)* of the original image. We want the estimate to be as close as possible to the original image. The more we know about h and ɖ , the closer f(x,y) will be to f'(x,y).

If it is a linear position invariant process, then degraded image is given in the spatial domain by

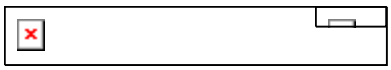
$$g(x,y)=f(x,y)*h(x,y)+\eta(x,y)$$

h(x,y) is spatial representation of degradation function and symbol * represents convolution. In frequency domain we may write this equation as

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

The terms in the capital letters are the Fourier Transform of the corresponding terms in the spatial domain.

The image restoration process can be achieved by inversing the image degradation process, i.e.,



Where 1/H(u,v)is the inverse filter, and $\acute{u}(u,v)$ is the recovered image. Although the concept is relatively simple, the actual implementation is difficult to achieve, as one requires prior knowledge or identifications of the unknown degradation function h(x,y) and the unknown noise source n(x,y.)

In the following sections, common noise models and method of estimating the degradation function are presented

Noise Models

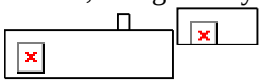
The principal source of noise in digital images arises during image acquisition and /or transmission. The performance of imaging sensors is affected by a variety of factors, such as environmental conditions during image acquisition and by the quality of the sensing elements themselves. Images are corrupted during transmission principally due to interference in the channels used for transmission. Since main sources of noise presented in digital images are resulted from atmospheric disturbance and image sensor circuitry, following assumptions can be made:

1The noise model is spatial invariant, i.e., independent of spatial location.
2The noise model is uncorrelated with the object function.

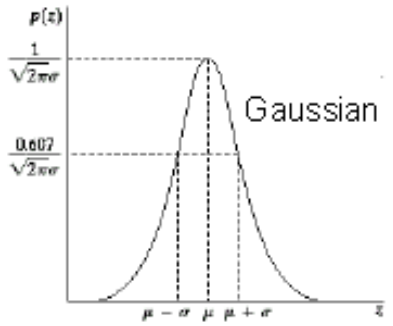
I. Gaussian Noise

These noise models are used frequently in practices because of its tractability in both spatial and frequency domain.

The PDF of Gaussian random variable, z is given by

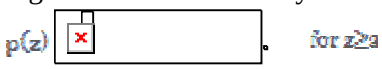


z= gray level
μ= mean of average value of z
ůů= standard deviation

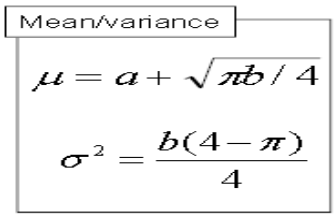


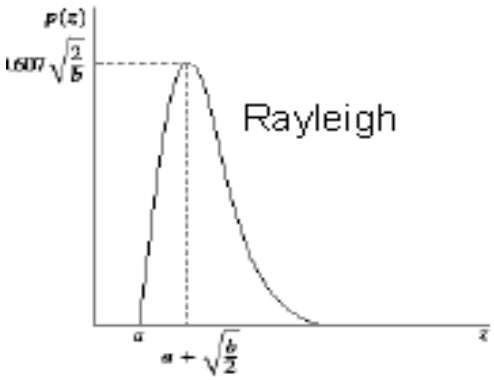
.2 Rayleigh Noise

Unlike Gaussian distribution, the Rayleigh distribution is no symmetric. It is given by the formula.



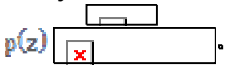
The mean and variance of this density





3 Erlang (gamma) Noise

The PDF of Erlang noise is given by

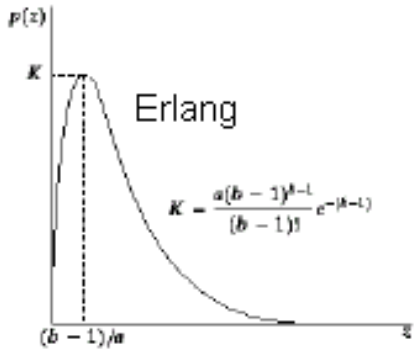


The mean and variance of this noise is

Mean/variance

$$\mu = \frac{b}{a}$$

$$\sigma^2 = \frac{b}{a^2}$$



Its shape is similar to Rayleigh disruption.
This equation is referred to as gamma density it is correct only when the denominator is the gamma function.

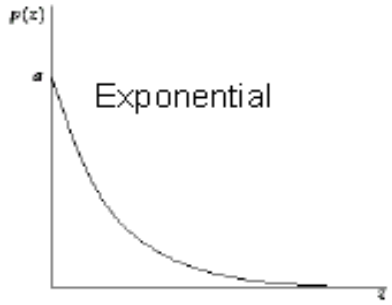
4. Exponential Noise

Exponential distribution has an exponential shape.
The PDF of exponential noise is given as

$$p(z) = ae^{-az}, \quad \text{for } z \geq 0$$

Where a>0

It is a special case of Erlang with b=1



5. Uniform Noise

The PDF of uniform noise is given by

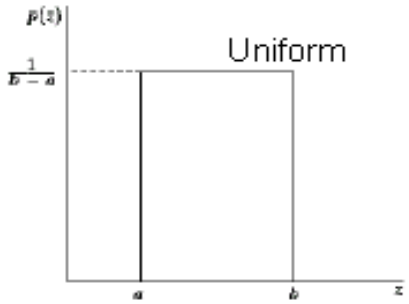
$$p(z) = \begin{cases} \frac{1}{(b-a)} & a \leq z \leq b \\ 0 & \text{otherwise} \end{cases}$$

The mean of this density function is given by

Mean/variance

$$\mu = \frac{a+b}{2}$$

$$\sigma^2 = \frac{(b-a)^2}{12}$$

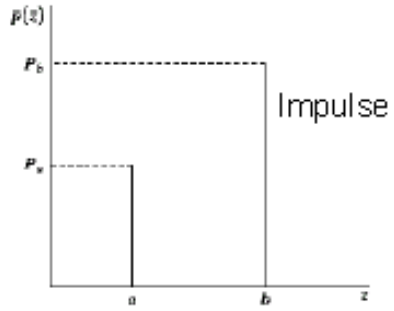


6. Impulse (Salt and Pepper)Noise

In this case, the noise is signal dependent, and is multiplied to the image.
The PDF of bipolar (impulse) noise is given by

$$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$$

If b>a, gray level b will appear as a light dot in image.
Level a will appear like a dark dot.



Restoration In the Presence of Noise Only-Spatial Filtering

When the only degradation present in an image is noise, i.e.

$$g(x,y) = f(x,y) + d(x,y)$$
or
$$G(u,v) = F(u,v) + N(u,v)$$

The noise terms are unknown so subtracting them from g(x,y) or G(u,v) is not a realistic approach. In the case of periodic noise it is possible to estimate N(u,v) from the spectrum G(u,v). So N(u,v) can be subtracted from G(u,v) to obtain an estimate of original image. Spatial filtering can be done when only additive noise is present.

The following techniques can be used to reduce the noise effect:

Mean Filter

Arithmetic Mean Filter

It is the simplest mean filter. Let S_{xy} represents the set of coordinates in the sub image of size m*n centered at point (x,y). The arithmetic mean filter computes the average value of the corrupted image g(x,y) in the area defined by S_{xy}. The value of the restored image f at any point (x,y) is the arithmetic mean computed using the pixels in the region defined by S_{xy}.

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t) \in Sxy} g(s,t)$$

This operation can be using a convolution mask in which all coefficients have value 1/mn
A mean filter smoothes local variations in image Noise is reduced as a result of blurring. For every pixel in the image, the pixel value is replaced by the mean value of its neighboring pixels (with a weight 1/(MM). This will resulted in a smoothing effect in the image.

Geometric mean filter
An image restored using a geometric mean filter is given by the expression

$$\hat{f}(x,y) = \left(\prod_{(s,t) \in Sxy} g(s,t) \right)^{1/mn}$$

Here, each restored pixel is given by the product of the pixel in the subimage window, raised to the power 1/mn. A geometric mean filters but it to loose image details in the process.

Harmonic mean filter
The harmonic mean filtering operation is given by the expression

$$\hat{f}(x,y) = \sum_{(s,t) \in Sxy} g(s,t)^{Q+1} \Big/ \sum_{(s,t) \in Sxy} g(s,t)^{Q}$$

The harmonic mean filter works well for salt noise but fails for pepper noise. It does well with Gaussian noise also.
Order statistics filter
Order statistics filters are spatial filters whose response is based on ordering the pixel contained in the image area encompassed by the filter.
The response of the filter at any point is determined by the ranking result.

Median filter
It is the best order statistic filter; it replaces the value of a pixel by the median of gray levels in the Neighborhood of the pixel.

$$\hat{f}(x,y) = \underset{(s,t) \in Sxy}{\text{median}} \{g(s,t)\}$$

The original of the pixel is included in the computation of the median of the filter are quite possible because for certain types of random noise, the provide excellent noise reduction capabilities with considerably less blurring then smoothing filters of similar size. These are effective for bipolar and unipolor impulse noise.

Max and Min Filters
Using the l00th percentile of ranked set of numbers is called the max filter and is given by the equation

$$\hat{f}(x,y) = \max_{(s,t) \in Sxy} \{g(s,t)\}$$

It is used for finding the brightest point in an image. Pepper noise in the image has very low values, it is reduced by max filter using the max selection process in the sublimated area sky.

The 0^{th} percentile filter is min filter

$$\hat{f}(x,y) = \min_{(s,t)\in Sxy} \{g(s,t)\}$$

This filter is useful for flinging the darkest point in image. Also, it reduces salt noise of the min operation.

a. Midpoint Filter

The midpoint filter simply computes the midpoint between the maximum and minimum values in the area encompassed by the filter

$$\hat{f}(x,y) = \left(\max_{(s,t)\in Sxy} \{g(s,t)\} + \min_{(s,t)\in Sxy} \{g(s,t)\} \right) / 2$$

It comeliness the order statistics and averaging .This filter works best for randomly distributed noise like Gaussian or uniform noise.

Periodic Noise By Frequency Domain Filtering

These types of filters are used for this purpose-

Band Reject Filters

It removes a band of frequencies about the origin of the Fourier transformer.

Ideal Band reject Filter

An ideal band reject filter is given by the expression

$$H(u,v) = \begin{cases} 1 & \text{if} \quad D(u,v) < D_0 - W/2 \\ 0 & \text{if} \quad D_0 - W/2 \le D(u,v) \le D_0 + W/2 \\ 1 & \text{if} \quad D(u,v) > D_0 + W/2 \end{cases}$$

D(u,v)- the distance from the origin of the centered frequency rectangle.
W- the width of the band
Do- the radial center of the frequency rectangle.

Butterworth Band reject Filter

$$H(u,v) = 1 \Big/ \left[1 + \left(\frac{D(u,v)W}{D^2(u,v) - D_0^2} \right)^{2n} \right]$$

Gaussian Band reject Filter

$$H(u,v) = 1 - \exp\left[-\frac{1}{2} \left(\frac{D^2(u,v) - D_0^2}{D(u,v)W} \right)^2 \right]$$

These filters are mostly used when the location of noise component in the frequency domain is known. Sinusoidal noise can be easily removed by using these kinds of filters because it shows two impulses that are mirror images of each other about the origin. Of the frequency transform.

Band Pass Filters

The function of a band pass filter is opposite to that of a band reject filter It allows a specific frequency band of the image to be passed and blocks the rest of frequencies.
The transfer function of a band pass filter can be obtained from a corresponding band reject filter with transfer function $H_{br}(u,v)$ by using the equation-

$$H_{BP}(u,v) = 1 - H_{BR}(u,v)$$

These filters cannot be applied directly on an image because it may remove too much details of an image but these are effective in isolating the effect of an image of selected frequency bands.

Notch Filters

This type of filters rejects frequencies I predefined in neighborhood above a centre frequency These filters are symmetric about origin in the Fourier transform the transfer function of ideal notch reject filter of radius do with centre at (é) and by symmetry at (é é) is

$$H(u,v) = \begin{cases} 0 & \text{if} \quad D_1(u,v) \le D_0 \text{ or } D_2(u,v) \le D_0 \\ 1 & \text{otherwise} \end{cases}$$

Where

$$D_1(u,v) = \sqrt{(u - M/2 - u_0)^2 + (v - N/2 - v_0)^2}$$
$$D_2(u,v) = \sqrt{(u - M/2 + u_0)^2 + (v - N/2 + v_0)^2}$$

Butterworth notch reject filter of order n is given by

$$H(u,v) = 1 - \exp\left[-\frac{1}{2}\left(\frac{D_1(u,v) D_2(u,v)}{D_0^2} \right) \right]$$

A Gaussian notch reject filter has the fauna

$$H(u,v) = 1 \Big/ \left[1 + \left(\frac{D_0^2}{D_1(u,v) D_2(u,v)} \right)^n \right]$$

These filter become high pass rather than suppress. The frequencies contained in the notch areas. These filters will perform exactly the opposite function as the notch reject filter.
The transfer function of this filter may be given as

$$Hnp(u,v) = 1 - Hnr(u,v)$$

$H_{np}(u,v)$- transfer function of the pass filter
$H_{nr}(u,v)$- transfer function of a notch reject filter

Minimum Mean Square Error (Wiener) Filtering

This filter incorporates both degradation function and statistical behavior of noise into the restoration process.
The main concept behind this approach is that the images and noise are considered as random variables and the objective is to find an estimate f of the uncorrupted image f such that the mean sequence error between then is minimized.

$$\hat{f}(x) = \sum_{s=-\infty}^{\infty} h_W(x-s)g(s),$$

This error measure is given by

$$e^2 = E\{[f(x) - \hat{f}(x)]^2\} = \min$$

Where e( ) is the expected value of the argument
Assuming that the noise and the image are uncorrelated (means zero average value) one or other has zero mean values
The minimum error function of the above expression is given in the frequency é é .. is given by the expression.

$$H_W(u,v) = \frac{H^*(u,v)S_{ff}(u,v)}{|H(u,v)|^2 S_{ff}(u,v) + S_{nn}(u,v)} = \frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + S_{nn}(u,v)/S_{ff}(u,v)}$$

Product of a complex quantity with its conjugate is equal to the magnitude of é é complex quantity squared. This result is known as wiener Filter The filter was named so because of the name of its inventor N Wiener. The term in the bracket is known as minimum mean square error filter or least square error filter.

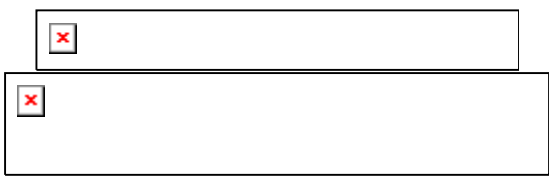
H*(u,v)-degradation function .
H*(u,v)-complex conjugate of H(u,v)
H(u,v) H(u,v)
Sn(u,v)=IN(u,v)I^2- power spectrum of the noise
Sf(u,v)=IF(u,v)^2- power spectrum of the underrated image
H(u,v)=Fourier transformer of the degraded function
G(u,v)=Fourier transformer of the degraded image

The restored image in the spatial domain is given by the inverse Fourier transformed of the frequency domain estimate F(u,v).
Mean square error in statistical form can be approveiment by the function

$$H_W(u,v) = \frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + K}$$

Inverse Filtering

It is a process of restoring an image degraded by a degradation function H. This function can be obtained by any method.
The simplest approach to restoration is direct, inverse filtering.
Inverse filtering provides an estimate F(u,v) of the transform of the original image simply by during the transform of the degraded image G(u,v) by the degradation function.



1.Define image compression. Explain about the redundancies in a digital image.

The term data compression refers to the process of reducing the amount of data required to represent a given quantity of information. A clear distinction must be made between data and information. They are not synonymous. In fact, data are the means by which information is conveyed. Various amounts of data may be used to represent the same amount of information. Such might be the case, for example, if a long-winded individual and someone who is short and to the point were to relate the same story. Here, the information of interest is the story; words are the data used to relate the information. If the two individuals use a different number of words to tell the same basic story, two different versions of the story are created, and at least one includes nonessential data. That is, it contains data (or words) that either provide no relevant information or simply restate that which is already known. It is thus said to contain data redundancy.

Data redundancy is a central issue in digital image compression. It is not an abstract concept but a mathematically quantifiable entity. If n_1 and n_2 denote the number of information-carrying units in two data sets that represent the same information, the relative data redundancy R_D of the first data set (the one characterized by n_1) can be defined as

$$R_D = 1 - \frac{1}{C_R}$$

where C_R , commonly called the compression ratio, is

$$C_R = \frac{n_1}{n_2}.$$

For the case $n_2 = n_1$, $C_R = 1$ and $R_D = 0$, indicating that (relative to the second data set) the first representation of the information contains no redundant data. When $n_2 << n_1$, $C_R \to \infty$ and $R_D \to 1$, implying significant compression and highly redundant data. Finally, when $n_2 >> n_1$, $C_R \to 0$ and $R_D \to -\infty$, indicating that the second data set contains much more data than the original representation. This, of course, is the normally undesirable case of data expansion. In general, C_R and R_D lie in the open intervals $(0, \infty)$ and $(-\infty, 1)$, respectively. A practical compression ratio, such as 10 (or 10:1), means that the first data set has 10 information carrying
units (say, bits) for every 1 unit in the second or compressed data set. The corresponding redundancy of 0.9 implies that 90% of the data in the first data set is redundant.

In digital image compression, three basic data redundancies can be identified and exploited: **coding redundancy**, **interpixel redundancy**, and **psychovisual redundancy**. Data compression is achieved when one or more of these redundancies are reduced or eliminated.

Coding Redundancy:

In this, we utilize formulation to show how the gray-level histogram of an image also can provide a great deal of insight into the construction of codes to reduce the amount of data used to represent it.

Let us assume, once again, that a discrete random variable r_k in the interval [0, 1] represents the gray levels of an image and that each r_k occurs with probability $p_r(r_k)$.

$$p_r(r_k) = \frac{n_k}{n} \qquad k = 0, 1, 2, \ldots, L-1$$

where L is the number of gray levels, n_k is the number of times that the kth gray level appears in the image, and n is the total number of pixels in the image. If the number of bits used to represent each value of r_k is $l(r_k)$, then the average number of bits required to represent each pixel is

$$L_{avg} = \sum_{k=0}^{L-1} l(r_k) p_r(r_k).$$

That is, the average length of the code words assigned to the various gray-level values is found by summing the product of the number of bits used to represent each gray level and the probability that the gray level occurs. Thus the total number of bits required to code an M X N image is MNL_{avg}.

Interpixel Redundancy:

Consider the images shown in Figs. 1.1(a) and (b). As Figs. 1.1(c) and (d) show, these images have virtually identical histograms. Note also that both histograms are trimodal, indicating the presence of three dominant ranges of gray-level values. Because the gray levels in these images are not equally probable, variable-length coding can be used to reduce the coding redundancy that would result from a straight or natural binary encoding of their pixels. The coding process, however, would not alter the level of correlation between the pixels within the images. In other words, the codes used to represent the gray levels of each image have nothing to do with the correlation between pixels. These correlations result from the structural or geometric relationships between the objects in the image.

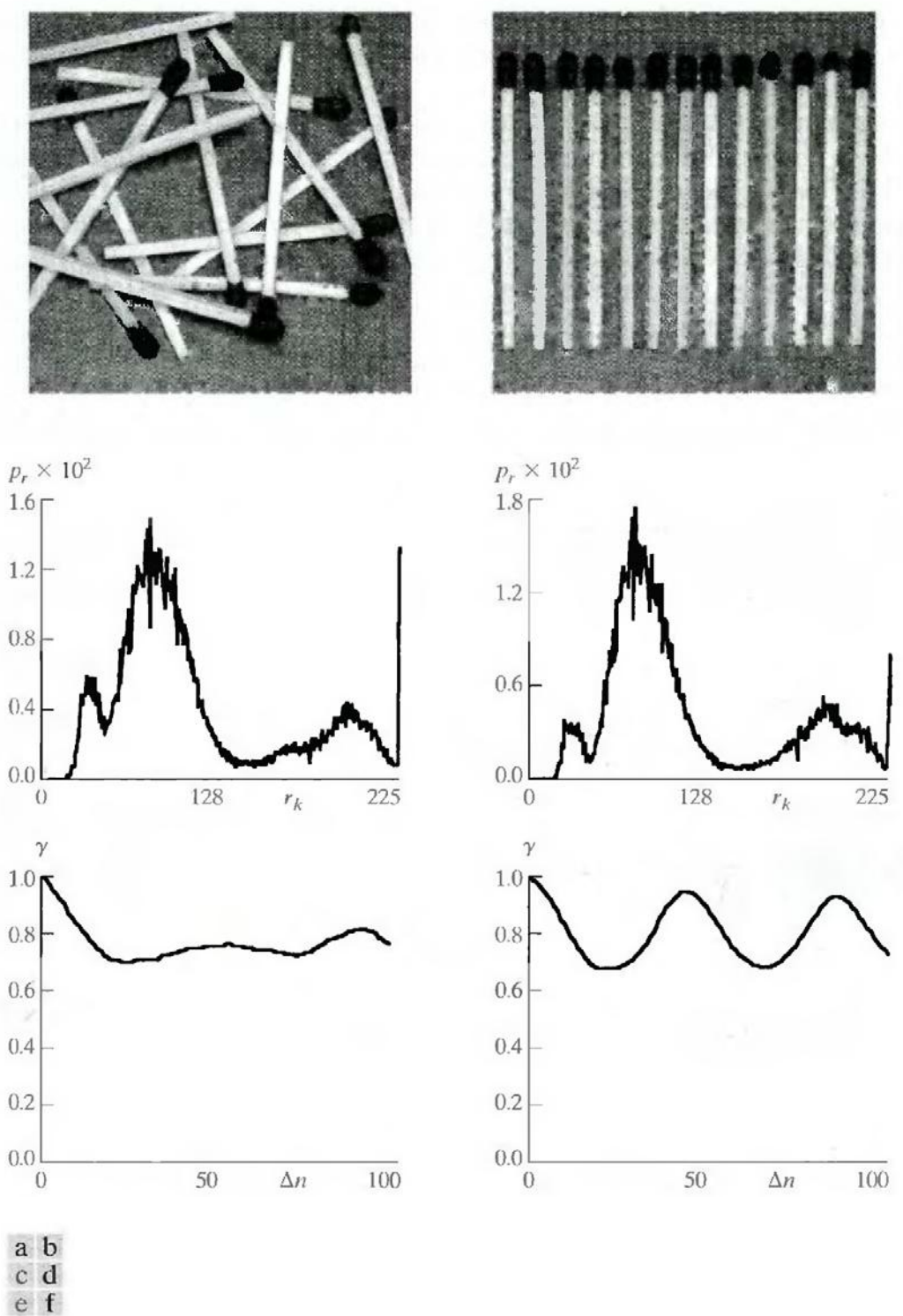


Fig.1.1 Two images and their gray-level histograms and normalized autocorrelation coefficients along one line.

Figures 1.1(e) and (f) show the respective autocorrelation coefficients computed along one line of each image.

$$\gamma(\Delta n) = \frac{A(\Delta n)}{A(0)}$$

where

$$A(\Delta n) = \frac{1}{N - \Delta n} \sum_{y=0}^{N-1-\Delta n} f(x, y) f(x, y + \Delta n).$$

The scaling factor in Eq. above accounts for the varying number of sum terms that arise for each integer value of Δn. Of course, Δn must be strictly less than N, the number of pixels on a line. The variable x is the coordinate of the line used in the computation. Note the dramatic difference between the shape of the functions shown in Figs. 1.1(e) and (f). Their shapes can be qualitatively related to the structure in the images in Figs. 1.1(a) and (b).This relationship is particularly noticeable in Fig. 1.1 (f), where the high correlation between pixels separated by 45 and 90 samples can be directly related to the spacing between the vertically oriented matches of Fig. 1.1(b). In addition, the adjacent pixels of both images are highly correlated. When Δn is 1, γ is 0.9922 and 0.9928 for the images of Figs. 1.1 (a) and (b), respectively. These values are typical of most properly sampled television images.

These illustrations reflect another important form of data redundancy—one directly related to the interpixel correlations within an image. Because the value of any given pixel can be reasonably predicted from the value of its neighbors, the information carried by individual pixels is relatively small. Much of the visual contribution of a single pixel to an image is redundant; it could have been guessed on the basis of the values of its neighbors. A variety of names, including spatial redundancy, geometric redundancy, and interframe redundancy, have been coined to refer to these interpixel dependencies. We use the term interpixel redundancy to encompass them all.

In order to reduce the interpixel redundancies in an image, the 2-D pixel array normally used for human viewing and interpretation must be transformed into a more efficient (but usually "nonvisual") format. For example, the differences between adjacent pixels can be used to represent an image. Transformations of this type (that is, those that remove interpixel redundancy) are referred to as mappings. They are called reversible mappings if the original image elements can be reconstructed from the transformed data set.

Psychovisual Redundancy:

The brightness of a region, as perceived by the eye, depends on factors other than simply the light reflected by the region. For example, intensity variations (Mach bands) can be perceived in an area of constant intensity. Such phenomena result from the fact that the eye does not respond with equal sensitivity to all visual information. Certain information simply has less relative importance than other information in normal visual processing. This information is said to be psychovisually redundant. It can be eliminated without significantly impairing the quality of image perception.

That psychovisual redundancies exist should not come as a surprise, because human perception of the information in an image normally does not involve quantitative analysis of every pixel value in the image. In general, an observer searches for distinguishing features such as edges or textural regions and mentally combines them into recognizable groupings. The brain then correlates these groupings with prior knowledge in order to complete the image interpretation process. Psychovisual redundancy is fundamentally different from the redundancies discussed earlier. Unlike coding and interpixel redundancy, psychovisual redundancy is associated with real or quantifiable visual information. Its elimination is possible only because the information itself is not essential for normal visual processing. Since the elimination of psychovisually redundant data results in a loss of quantitative information, it is commonly referred to as quantization.

This terminology is consistent with normal usage of the word, which generally means the mapping of a broad range of input values to a limited number of output values. As it is an irreversible operation (visual information is lost), quantization results in lossy data compression.

2. Explain about fidelity criterion.

The removal of psychovisually redundant data results in a loss of real or quantitative visual information. Because information of interest may be lost, a repeatable or reproducible means of quantifying the nature and extent of information loss is highly desirable. Two general classes of criteria are used as the basis for such an assessment:

A) Objective fidelity criteria and

B) Subjective fidelity criteria.

When the level of information loss can be expressed as a function of the original or input image and the compressed and subsequently decompressed output image, it is said to be based on an objective fidelity criterion. A good example is the root-mean-square (rms) error between an input and output image. Let f(x, y) represent an input image and let f(x, y) denote an estimate or approximation of f(x, y) that results from compressing and subsequently decompressing the input. For any value of x and y, the error e(x, y) between f (x, y) and f^ (x, y) can be defined as

$$e(x, y) = \hat{f}(x, y) - f(x, y)$$

so that the total error between the two images is

$$\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[\hat{f}(x, y) - f(x, y)\right]$$

where the images are of size M X N. The root-mean-square error, e_{rms}, between f(x, y) and f^(x, y) then is the square root of the squared error averaged over the M X N array, or

$$e_{rms} = \left[\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[\hat{f}(x, y) - f(x, y)\right]^2\right]^{1/2}$$

A closely related objective fidelity criterion is the mean-square signal-to-noise ratio of the compressed-decompressed image. If f^ (x, y) is considered to be the sum of the original image f(x, y) and a noise signal e(x, y), the mean-square signal-to-noise ratio of the output image, denoted SNR_{rms}, is

$$SNR_{ms} = \frac{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\hat{f}(x, y)^2}{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[\hat{f}(x, y) - f(x, y)\right]^2}.$$

The rms value of the signal-to-noise ratio, denoted SNR_{rms}, is obtained by taking the square root of Eq. above.

Although objective fidelity criteria offer a simple and convenient mechanism for evaluating information loss, most decompressed images ultimately are viewed by humans. Consequently, measuring image quality by the subjective evaluations of a human observer often is more appropriate. This can be accomplished by showing a "typical" decompressed image to an appropriate cross section of viewers and averaging their evaluations. The evaluations may be made using an absolute rating scale or by means of side-by-side comparisons of f(x, y) and f^(x,

y).

3. Explain about image compression models.

Fig. 3.1 shows, a compression system consists of two distinct structural blocks: an encoder and a decoder. An input image f(x, y) is fed into the encoder, which creates a set of symbols from the input data. After transmission over the channel, the encoded representation is fed to the decoder, where a reconstructed output image f^(x, y) is generated. In general, f^(x, y) may or may not be an exact replica of f(x, y). If it is, the system is error free or information preserving; if not, some level of distortion is present in the reconstructed image. Both the encoder and decoder shown in Fig. 3.1 consist of two relatively independent functions or subblocks. The encoder is made up of a source encoder, which removes input redundancies, and a channel encoder, which increases the noise immunity of the source encoder's output. As would be expected, the decoder includes a channel decoder followed by a source decoder. If the channel between the encoder and decoder is noise free (not prone to error), the channel encoder and decoder are omitted, and the general encoder and decoder become the source encoder and decoder, respectively.

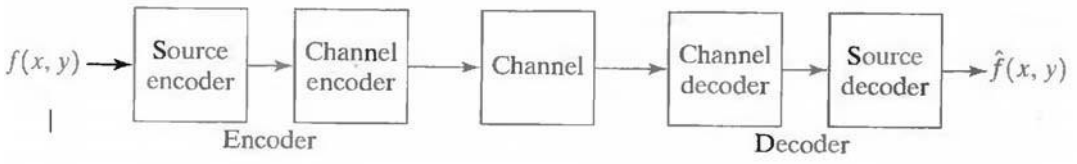
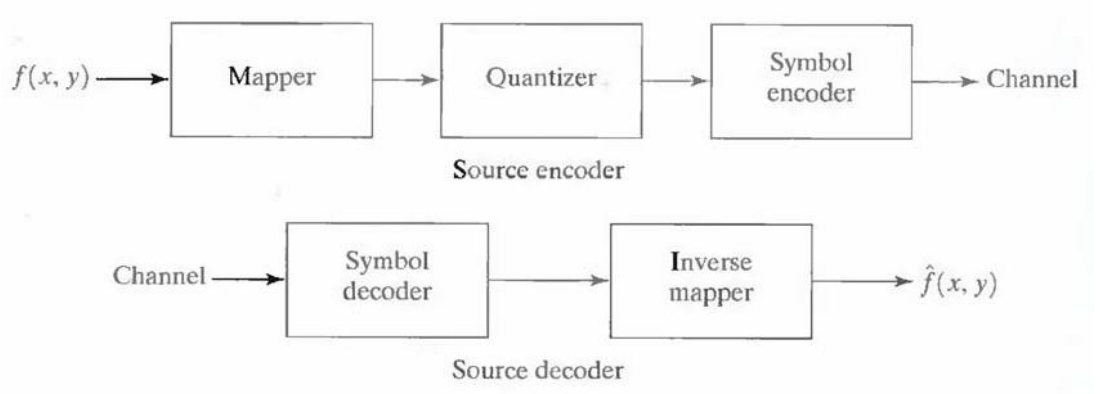


Fig.3.1 A general compression system model

The Source Encoder and Decoder:

The source encoder is responsible for reducing or eliminating any coding, interpixel, or psychovisual redundancies in the input image. The specific application and associated fidelity requirements dictate the best encoding approach to use in any given situation. Normally, the approach can be modeled by a series of three independent operations. As Fig. 3.2 (a) shows, each operation is designed to reduce one of the three redundancies. Figure 3.2 (b) depicts the corresponding source decoder. In the first stage of the source encoding process, the mapper transforms the input data into a (usually nonvisual) format designed to reduce interpixel redundancies in the input image. This operation generally is reversible and may or may not reduce directly the amount of data required to represent the image.



a
b

Fig.3.2 (a) Source encoder and (b) source decoder model

Run-length coding is an example of a mapping that directly results in data compression in this initial stage of the overall source encoding process. The representation of an image by a set of transform coefficients is an example of the opposite case. Here, the mapper transforms the image into an array of coefficients, making its interpixel redundancies more accessible for compression in later stages of the encoding process.

The second stage, or quantizer block in Fig. 3.2 (a), reduces the accuracy of the mapper's output in accordance with some preestablished fidelity criterion. This stage reduces the psychovisual redundancies of the input image. This operation is irreversible. Thus it must be omitted when error-free compression is desired.

In the third and final stage of the source encoding process, the symbol coder creates a fixed- or variable-length code to represent the quantizer output and maps the output in accordance with the code. The term symbol coder distinguishes this coding operation from the overall source encoding process. In most cases, a variable-length code is used to represent the mapped and quantized data set. It assigns the shortest code words to the most frequently occurring output values and thus reduces coding redundancy. The operation, of course, is reversible. Upon completion of the symbol coding step, the input image has been processed to remove each of the three redundancies.

Figure 3.2(a) shows the source encoding process as three successive operations, but all three operations are not necessarily included in every compression system. Recall, for example, that the quantizer must be omitted when error-free compression is desired. In addition, some compression techniques normally are modeled by merging blocks that are physically separate in

Fig. 3.2(a). In the predictive compression systems, for instance, the mapper and quantizer are often represented by a single block, which simultaneously performs both operations.

The source decoder shown in Fig. 3.2(b) contains only two components: a symbol decoder and an inverse mapper. These blocks perform, in reverse order, the inverse operations of the source encoder's symbol encoder and mapper blocks. Because quantization results in irreversible information loss, an inverse quantizer block is not included in the general source decoder model shown in Fig. 3.2(b).

The Channel Encoder and Decoder:

The channel encoder and decoder play an important role in the overall encoding-decoding process when the channel of Fig. 3.1 is noisy or prone to error. They are designed to reduce the impact of channel noise by inserting a controlled form of redundancy into the source encoded data. As the output of the source encoder contains little redundancy, it would be highly sensitive to transmission noise without the addition of this "controlled redundancy." One of the most useful channel encoding techniques was devised by R. W. Hamming (Hamming [1950]). It is based on appending enough bits to the data being encoded to ensure that some minimum number of bits must change between valid code words. Hamming showed, for example, that if 3 bits of redundancy are added to a 4-bit word, so that the distance between any two valid code words is 3, all single-bit errors can be detected and corrected. (By appending additional bits of redundancy, multiple-bit errors can be detected and corrected.) The 7-bit Hamming (7, 4) code word $h_1, h_2, h_3 \ldots, h_6, h_7$ associated with a 4-bit binary number $b_3b_2b_1b_0$ is

$$
\begin{aligned}
h_1 &= b_3 \oplus b_2 \oplus b_0 \qquad & h_3 &= b_3 \\
h_2 &= b_3 \oplus b_1 \oplus b_0 \qquad & h_5 &= b_2 \\
h_4 &= b_2 \oplus b_1 \oplus b_0 \qquad & h_6 &= b_1 \\
& & h_7 &= b_0
\end{aligned}
$$

where $\oplus$ denotes the exclusive OR operation. Note that bits h_1, h_2, and h_4 are even- parity bits for the bit fields $b_3 b_2 b_0$, $b_3b_1b_0$, and $b_2b_1b_0$, respectively. (Recall that a string of binary bits has even parity if the number of bits with a value of 1 is even.) To decode a Hamming encoded result, the channel decoder must check the encoded value for odd parity over the bit fields in which even parity was previously established. A single-bit error is indicated by a nonzero parity word $c_4c_2c_1$, where

$$c_1 = h_1 \oplus h_3 \oplus h_5 \oplus h_7$$
$$c_2 = h_2 \oplus h_3 \oplus h_6 \oplus h_7$$
$$c_4 = h_4 \oplus h_5 \oplus h_6 \oplus h_7.$$

If a nonzero value is found, the decoder simply complements the code word bit position indicated by the parity word. The decoded binary value is then extracted from the corrected code word as $h_3h_5h_6h_7$.

4. Explain a method of generating variable length codes with an example.

Variable-Length Coding:

The simplest approach to error-free image compression is to reduce only coding redundancy. Coding redundancy normally is present in any natural binary encoding of the gray levels in an image. It can be eliminated by coding the gray levels. To do so requires construction of a variable-length code that assigns the shortest possible code words to the most probable gray levels. Here, we examine several optimal and near optimal techniques for constructing such a code. These techniques are formulated in the language of information theory. In practice, the source symbols may be either the gray levels of an image or the output of a gray-level mapping operation (pixel differences, run lengths, and so on).

Huffman coding:

The most popular technique for removing coding redundancy is due to Huffman (Huffman [1952]). When coding the symbols of an information source individually, Huffman coding yields the smallest possible number of code symbols per source symbol. In terms of the noiseless coding theorem, the resulting code is optimal for a fixed value of n, subject to the constraint that the source symbols be coded one at a time.

The first step in Huffman's approach is to create a series of source reductions by ordering the probabilities of the symbols under consideration and combining the lowest probability symbols into a single symbol that replaces them in the next source reduction. Figure 4.1 illustrates this process for binary coding (K-ary Huffman codes can also be constructed). At the far left, a hypothetical set of source symbols and their probabilities are ordered from top to bottom in terms of decreasing probability values. To form the first source reduction, the bottom two probabilities, 0.06 and 0.04, are combined to form a "compound symbol" with probability 0.1. This compound symbol and its associated probability are placed in the first source reduction column so that the

probabilities of the reduced source are also ordered from the most to the least probable. This process is then repeated until a reduced source with two symbols (at the far right) is reached.

The second step in Huffman's procedure is to code each reduced source, starting with the smallest source and working back to the original source. The minimal length binary code for a two-symbol source, of course, is the symbols 0 and 1. As Fig. 4.2 shows, these symbols are assigned to the two symbols on the right (the assignment is arbitrary; reversing the order of the 0 and 1 would work just as well). As the reduced source symbol with probability 0.6 was generated by combining two symbols in the reduced source to its left, the 0 used to code it is now assigned to both of these symbols, and a 0 and 1 are arbitrarily

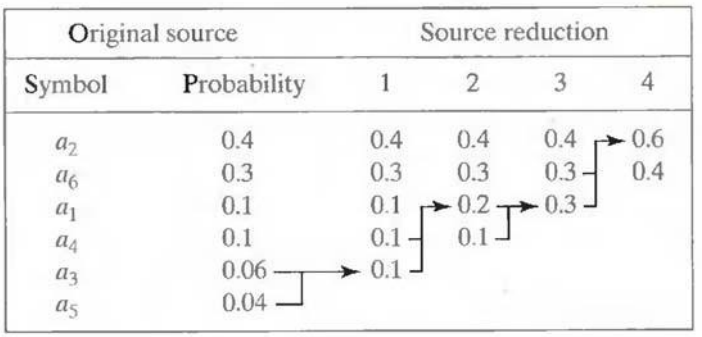


Fig.4.1 Huffman source reductions.

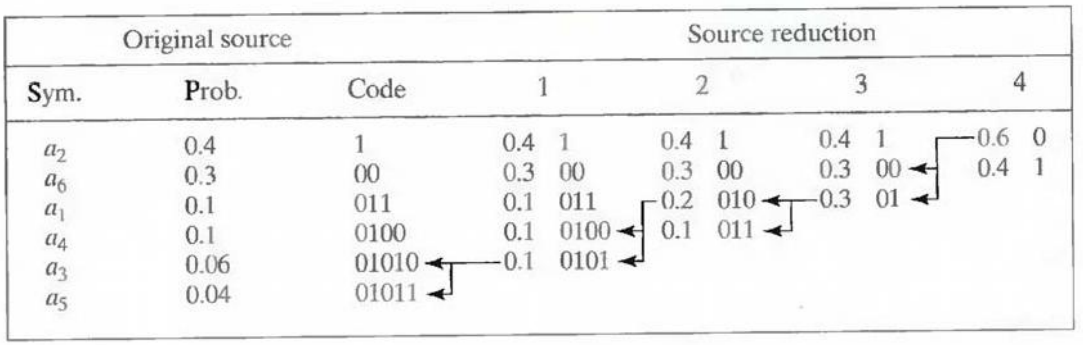


Fig.4.2 Huffman code assignment procedure.

appended to each to distinguish them from each other. This operation is then repeated for each reduced source until the original source is reached. The final code appears at the far left in Fig. 4.2. The average length of this code is

$$L_{avg} = (0.4)(1) + (0.3)(2) + (0.1)(3) + (0.1)(4) + (0.06)(5) + (0.04)(5)$$
$$= 2.2 \text{ bits/symbol}$$

and the entropy of the source is 2.14 bits/symbol. The resulting Huffman code efficiency is 0.973.

Huffman's procedure creates the optimal code for a set of symbols and probabilities subject to the constraint that the symbols be coded one at a time. After the code has been created, coding and/or decoding is accomplished in a simple lookup table manner. The code itself is an instantaneous uniquely decodable block code. It is called a block code because each source symbol is mapped into a fixed sequence of code symbols. It is instantaneous, because each code word in a string of code symbols can be decoded without referencing succeeding symbols. It is uniquely decodable, because any string of code symbols can be decoded in only one way. Thus, any string of Huffman encoded symbols can be decoded by examining the individual symbols of the string in a left to right manner. For the binary code of Fig. 4.2, a left-to-right scan of the encoded string 010100111100 reveals that the first valid code word is 01010, which is the code for symbol a_3 .The next valid code is 011, which corresponds to symbol a_1. Continuing in this manner reveals the completely decoded message to be $a_3a_1a_2a_2a_6$.

5. Explain arithmetic encoding process with an example.

Arithmetic coding:

Unlike the variable-length codes described previously, arithmetic coding generates nonblock codes. In arithmetic coding, which can be traced to the work of Elias, a one-to-one correspondence between source symbols and code words does not exist. Instead, an entire sequence of source symbols (or message) is assigned a single arithmetic code word. The code word itself defines an interval of real numbers between 0 and 1. As the number of symbols in the message increases, the interval used to represent it becomes smaller and the number of information units (say, bits) required to represent the interval becomes larger. Each symbol of the message reduces the size of the interval in accordance with its probability of occurrence. Because the technique does not require, as does Huffman's approach, that each source symbol translate into an integral number of code symbols (that is, that the symbols be coded one at a time), it achieves (but only in theory) the bound established by the noiseless coding theorem.

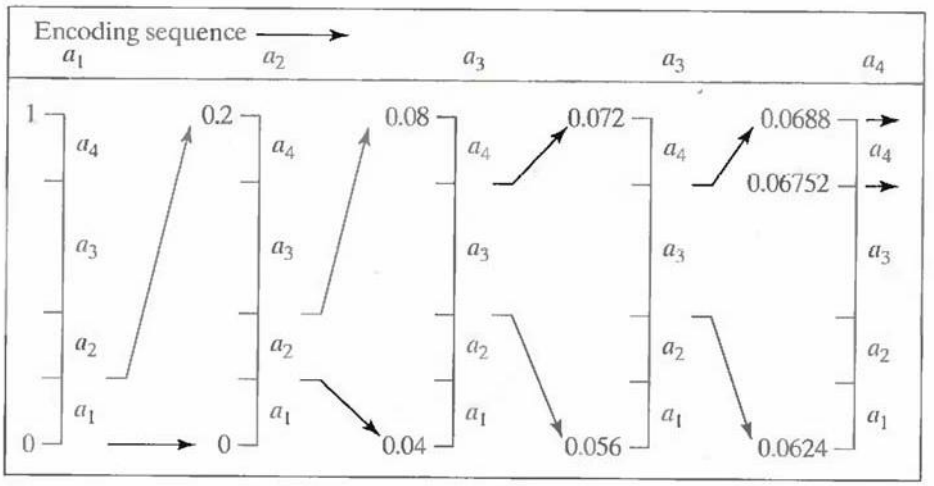


Fig.5.1 Arithmetic coding procedure

Figure 5.1 illustrates the basic arithmetic coding process. Here, a five-symbol sequence or message, $a_1a_2a_3a_3a_4$, from a four-symbol source is coded. At the start of the coding process, the message is assumed to occupy the entire half-open interval [0, 1). As Table 5.2 shows, this interval is initially subdivided into four regions based on the probabilities of each source symbol. Symbol ax, for example, is associated with subinterval [0, 0.2). Because it is the first symbol of the message being coded, the message interval is initially narrowed to [0, 0.2). Thus in Fig. 5.1 [0, 0.2) is expanded to the full height of the figure and its end points labeled by the values of the narrowed range. The narrowed range is then subdivided in accordance with the original source symbol probabilities and the process continues with the next message symbol.

Source Symbol	Probability	Initial Subinterval
a_1	0.2	[0.0, 0.2)
a_2	0.2	[0.2, 0.4)
a_3	0.4	[0.4, 0.8)
a_4	0.2	[0.8, 1.0)

Table 5.1 Arithmetic coding example

In this manner, symbol a_2 narrows the subinterval to [0.04, 0.08), a_3 further narrows it to [0.056, 0.072), and so on. The final message symbol, which must be reserved as a special end-of-

message indicator, narrows the range to [0.06752, 0.0688). Of course, any number within this subinterval—for example, 0.068—can be used to represent the message.

In the arithmetically coded message of Fig. 5.1, three decimal digits are used to represent the five-symbol message. This translates into 3/5 or 0.6 decimal digits per source symbol and compares favorably with the entropy of the source, which is 0.58 decimal digits or 10-ary units/symbol. As the length of the sequence being coded increases, the resulting arithmetic code approaches the bound established by the noiseless coding theorem.

In practice, two factors cause coding performance to fall short of the bound: (1) the addition of the end-of-message indicator that is needed to separate one message from another; and (2) the use of finite precision arithmetic. Practical implementations of arithmetic coding address the latter problem by introducing a scaling strategy and a rounding strategy (Langdon and Rissanen [1981]). The scaling strategy renormalizes each subinterval to the [0, 1) range before subdividing it in accordance with the symbol probabilities. The rounding strategy guarantees that the truncations associated with finite precision arithmetic do not prevent the coding subintervals from being represented accurately.

6. Explain LZW coding with an example.

LZW Coding:

The technique, called Lempel-Ziv-Welch (LZW) coding, assigns fixed-length code words to variable length sequences of source symbols but requires no a priori knowledge of the probability of occurrence of the symbols to be encoded. LZW compression has been integrated into a variety of mainstream imaging file formats, including the graphic interchange format (GIF), tagged image file format (TIFF), and the portable document format (PDF).

LZW coding is conceptually very simple (Welch [1984]). At the onset of the coding process, a codebook or "dictionary" containing the source symbols to be coded is constructed. For 8-bit monochrome images, the first 256 words of the dictionary are assigned to the gray values 0, 1, 2..., and 255. As the encoder sequentially examines the image's pixels, gray-level sequences that are not in the dictionary are placed in algorithmically determined (e.g., the next unused) locations. If the first two pixels of the image are white, for instance, sequence “255-255” might be assigned to location 256, the address following the locations reserved for gray levels 0 through 255. The next time that two consecutive white pixels are encountered, code word 256, the address of the location containing sequence 255-255, is used to represent them. If a 9-bit, 512-word dictionary is employed in the coding process, the original (8 + 8) bits that were

used to represent the two pixels are replaced by a single 9-bit code word. Cleary, the size of the dictionary is an important system parameter. If it is too small, the detection of matching gray-level sequences will be less likely; if it is too large, the size of the code words will adversely affect compression performance.

Consider the following 4 x 4, 8-bit image of a vertical edge:

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Table 6.1 details the steps involved in coding its 16 pixels. A 512-word dictionary with the following starting content is assumed:

Dictionary Location	Entry
0	0
1	1
⋮	⋮
255	255
256	—
⋮	⋮
511	—

Locations 256 through 511 are initially unused. The image is encoded by processing its pixels in a left-to-right, top-to-bottom manner. Each successive gray-level value is concatenated with a variable—column 1 of Table 6.1 —called the "currently recognized sequence." As can be seen, this variable is initially null or empty. The dictionary is searched for each concatenated sequence and if found, as was the case in the first row of the table, is replaced by the newly concatenated and recognized (i.e., located in the dictionary) sequence. This was done in column 1 of row 2.

Currently Recognized Sequence	Pixel Being Processed	Encoded Output	Dictionary Location (Code Word)	Dictionary Entry
	39			
39	39	39	256	39-39
39	126	39	257	39-126
126	126	126	258	126-126
126	39	126	259	126-39
39	39			
39-39	126	256	260	39-39-126
126	126			
126-126	39	258	261	126-126-39
39	39			
39-39	126			
39-39-126	126	260	262	39-39-126-126
126	39			
126-39	39	259	263	126-39-39
39	126			
39-126	126	257	264	39-126-126
126		126		

Table 6.1 LZW coding example

No output codes are generated, nor is the dictionary altered. If the concatenated sequence is not found, however, the address of the currently recognized sequence is output as the next encoded value, the concatenated but unrecognized sequence is added to the dictionary, and the currently recognized sequence is initialized to the current pixel value. This occurred in row 2 of the table. The last two columns detail the gray-level sequences that are added to the dictionary when scanning the entire 4 x 4 image. Nine additional code words are defined. At the conclusion of coding, the dictionary contains 265 code words and the LZW algorithm has successfully identified several repeating gray-level sequences—leveraging them to reduce the original 128-bit image lo 90 bits (i.e., 10 9-bit codes). The encoded output is obtained by reading the third column from top to bottom. The resulting compression ratio is 1.42:1.

A unique feature of the LZW coding just demonstrated is that the coding dictionary or code book is created while the data are being encoded. Remarkably, an LZW decoder builds an identical decompression dictionary as it decodes simultaneously the encoded data stream. . Although not needed in this example, most practical applications require a strategy for handling dictionary overflow. A simple solution is to flush or reinitialize the dictionary when

it becomes full and continue coding with a new initialized dictionary. A more complex option is to monitor compression performance and flush the dictionary when it becomes poor or unacceptable. Alternately, the least used dictionary entries can be tracked and replaced when necessary.

7. Explain the concept of bit plane coding method.

Bit-Plane Coding:

An effective technique for reducing an image's interpixel redundancies is to process the image's bit planes individually. The technique, called bit-plane coding, is based on the concept of decomposing a multilevel (monochrome or color) image into a series of binary images and compressing each binary image via one of several well-known binary compression methods.

Bit-plane decomposition:

The gray levels of an m-bit gray-scale image can be represented in the form of the base 2 polynomial

$$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \cdots + a_1 2^1 + a_0 2^0.$$

Based on this property, a simple method of decomposing the image into a collection of binary images is to separate the m coefficients of the polynomial into m 1-bit bit planes. The zeroth-order bit plane is generated by collecting the a_0 bits of each pixel, while the (m - 1) st-order bit plane contains the a_{m-1}, bits or coefficients. In general, each bit plane is numbered from 0 to m-1 and is constructed by setting its pixels equal to the values of the appropriate bits or polynomial coefficients from each pixel in the original image. The inherent disadvantage of this approach is that small changes in gray level can have a significant impact on the complexity of the bit planes. If a pixel of intensity 127 (01111111) is adjacent to a pixel of intensity 128 (10000000), for instance, every bit plane will contain a corresponding 0 to 1 (or 1 to 0) transition. For example, as the most significant bits of the two binary codes for 127 and 128 are different, bit plane 7 will contain a zero-valued pixel next to a pixel of value 1, creating a 0 to 1 (or 1 to 0) transition at that point.

An alternative decomposition approach (which reduces the effect of small gray-level variations) is to first represent the image by an m-bit Gray code. The m-bit Gray code $g_{m-1}...g_2g_1g_0$ that corresponds to the polynomial in Eq. above can be computed from

$$g_i = a_i \oplus a_{i+1} \qquad 0 \le i \le m-2$$
$$g_{m-1} = a_{m-1}.$$

Here, ⊗ denotes the exclusive OR operation. This code has the unique property that successive code words differ in only one bit position. Thus, small changes in gray level are less likely to affect all m bit planes. For instance, when gray levels 127 and 128 are adjacent, only the 7th bit plane will contain a 0 to 1 transition, because the Gray codes that correspond to 127 and 128 are 11000000 and 01000000, respectively.

8. Explain about lossless predictive coding.

Lossless Predictive Coding:

The error-free compression approach does not require decomposition of an image into a collection of bit planes. The approach, commonly referred to as lossless predictive coding, is based on eliminating the interpixel redundancies of closely spaced pixels by extracting and coding only the new information in each pixel. The new information of a pixel is defined as the difference between the actual and predicted value of that pixel.

Figure 8.1 shows the basic components of a lossless predictive coding system. The system consists of an encoder and a decoder, each containing an identical predictor. As each successive pixel of the input image, denoted f_n, is introduced to the encoder, the predictor generates the anticipated value of that pixel based on some number of past inputs. The output of the predictor is then rounded to the nearest integer, denoted $f\hat{}_n$ and used to form the difference or prediction error which is coded using a variable-length code (by the symbol encoder) to generate the next element of the compressed data stream.

$$e_n = f_n - \hat{f}_n,$$

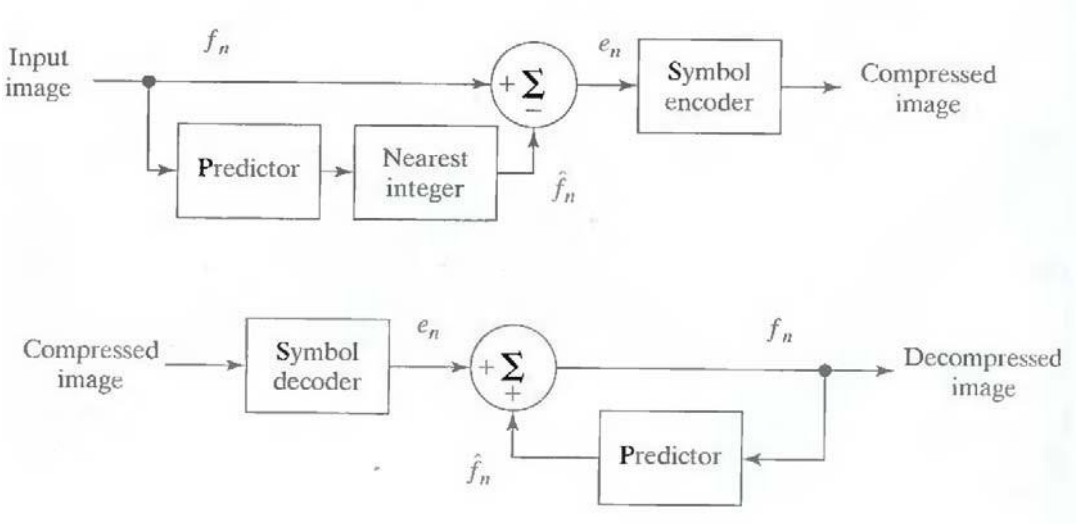


Fig.8.1 A lossless predictive coding model: (a) encoder; (b) decoder

The decoder of Fig. 8.1 (b) reconstructs en from the received variable-length code words and performs the inverse operation

$$f_n = e_n + \hat{f}_n.$$

Various local, global, and adaptive methods can be used to generate f^$_n$. In most cases, however, the prediction is formed by a linear combination of m previous pixels. That is,

$$\hat{f}_n = \text{round}\left[\sum_{i=1}^{m} \alpha_i f_{n-i}\right]$$

where m is the order of the linear predictor, round is a function used to denote the rounding or nearest integer operation, and the α_i, for i = 1,2,..., m are prediction coefficients. In raster scan applications, the subscript n indexes the predictor outputs in accordance with their time of occurrence. That is, f_n, f^$_n$ and e_n in Eqns. above could be replaced with the more explicit notation f (t), f^(t), and e (t), where t represents time. In other cases, n is used as an index on the spatial coordinates and/or frame number (in a time sequence of images) of an image. In 1-D linear predictive coding, for example, Eq. above can be written as

$$\hat{f}_n(x, y) = \text{round}\left[\sum_{i=1}^{m} \alpha_i f(x, y - i)\right]$$

where each subscripted variable is now expressed explicitly as a function of spatial coordinates x and y. The Eq. indicates that the 1-D linear prediction f(x, y) is a function of the previous pixels on the current line alone. In 2-D predictive coding, the prediction is a function of the previous pixels in a left-to-right, top-to-bottom scan of an image. In the 3-D case, it is based on these pixels and the previous pixels of preceding frames. Equation above cannot be evaluated for the first m pixels of each line, so these pixels must be coded by using other means (such as a Huffman code) and considered as an overhead of the predictive coding process. A similar comment applies to the higher-dimensional cases.

9. Explain about lossy predictive coding.

Lossy Predictive Coding:

In this type of coding, we add a quantizer to the lossless predictive model and examine the resulting trade-off between reconstruction accuracy and compression performance. As Fig.9 shows, the quantizer, which absorbs the nearest integer function of the error-free encoder, is inserted between the symbol encoder and the point at which the prediction error is formed. It maps the prediction error into a limited range of outputs, denoted $e^{\wedge}_n$ which establish the amount of compression and distortion associated with lossy predictive coding.

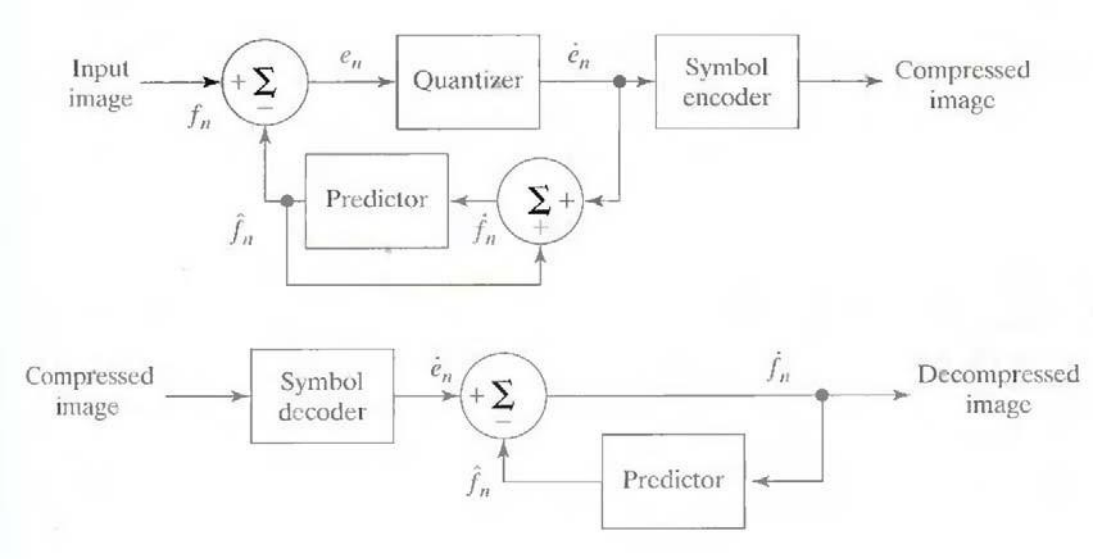


Fig. 9 A lossy predictive coding model: (a) encoder and (b) decoder.

In order to accommodate the insertion of the quantization step, the error-free encoder of figure must be altered so that the predictions generated by the encoder and decoder are equivalent. As Fig.9 (a) shows, this is accomplished by placing the lossy encoder's predictor within a feedback loop, where its input, denoted $\dot{f}_n$, is generated as a function of past predictions and the corresponding quantized errors. That is,

$$\dot{f}_n = \dot{e}_n + \hat{f}_n$$

This closed loop configuration prevents error buildup at the decoder's output. Note from Fig. 9 (b) that the output of the decoder also is given by the above Eqn.

Optimal predictors:

The optimal predictor used in most predictive coding applications minimizes the encoder's mean-square prediction error

$$E\{e_n^2\} = E\{[f_n - \hat{f}_n]^2\}$$

subject to the constraint that

$$\dot{f}_n = \dot{e}_n + \hat{f}_n \approx e_n + \hat{f}_n = f_n$$

and

$$\hat{f}_n = \sum_{i=1}^{m} \alpha_i f_{n-i}.$$

That is, the optimization criterion is chosen to minimize the mean-square prediction error, the quantization error is assumed to be negligible ($\dot{e}_n \approx e_n$), and the prediction is constrained to a linear combination of m previous pixels.1 These restrictions are not essential, but they simplify the analysis considerably and, at the same time, decrease the computational complexity of the predictor. The resulting predictive coding approach is referred to as differential pulse code modulation (DPCM).

10. Explain with a block diagram about transform coding system.

Transform Coding:

All the predictive coding techniques operate directly on the pixels of an image and thus are spatial domain methods. In this coding, we consider compression techniques that are based on modifying the transform of an image. In transform coding, a reversible, linear transform (such as the Fourier transform) is used to map the image into a set of transform coefficients, which are then quantized and coded. For most natural images, a significant number of the coefficients have small magnitudes and can be coarsely quantized (or discarded entirely) with little image distortion. A variety of transformations, including the discrete Fourier transform (DFT), can be used to transform the image data.

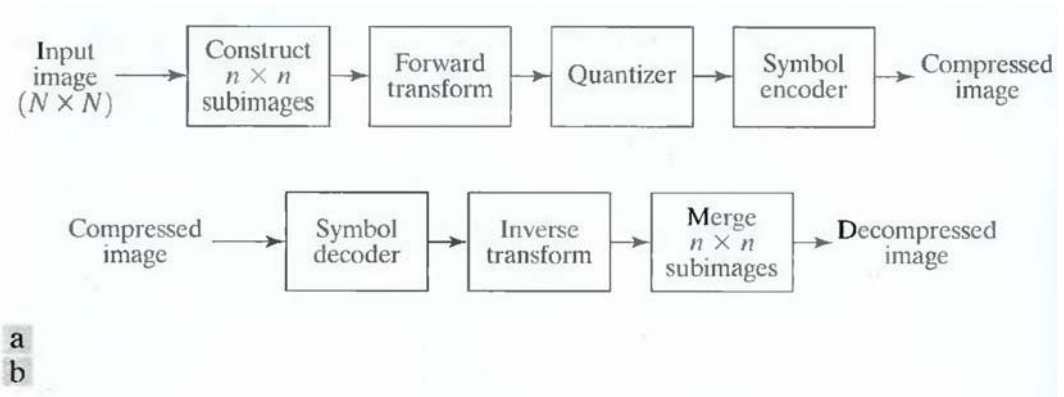


Fig. 10 A transform coding system: (a) encoder; (b) decoder.

Figure 10 shows a typical transform coding system. The decoder implements the inverse sequence of steps (with the exception of the quantization function) of the encoder, which performs four relatively straightforward operations: subimage decomposition, transformation, quantization, and coding. An N X N input image first is subdivided into subimages of size n X n, which are then transformed to generate $(N/n)^2$ subimage transform arrays, each of size n X n. The goal of the transformation process is to decorrelate the pixels of each subimage, or to pack as much information as possible into the smallest number of transform coefficients. The quantization stage then selectively eliminates or more coarsely quantizes the coefficients that carry the least information. These coefficients have the smallest impact on reconstructed subimage quality. The encoding process terminates by coding (normally using a variable-length code) the quantized coefficients. Any or all of the transform encoding steps can be adapted to

local image content, called adaptive transform coding, or fixed for all subimages, called nonadaptive transform coding.

11. Explain about wavelet coding.

Wavelet Coding:

The wavelet coding is based on the idea that the coefficients of a transform that decorrelates the pixels of an image can be coded more efficiently than the original pixels themselves. If the transform's basis functions—in this case wavelets—pack most of the important visual information into a small number of coefficients, the remaining coefficients can be quantized coarsely or truncated to zero with little image distortion.

Figure 11 shows a typical wavelet coding system. To encode a 2^J X 2^J image, an analyzing wavelet, Ψ, and minimum decomposition level, J - P, are selected and used to compute the image's discrete wavelet transform. If the wavelet has a complimentary scaling function φ, the fast wavelet transform can be used. In either case, the computed transform converts a large portion of the original image to horizontal, vertical, and diagonal decomposition coefficients with zero mean and Laplacian-like distributions.

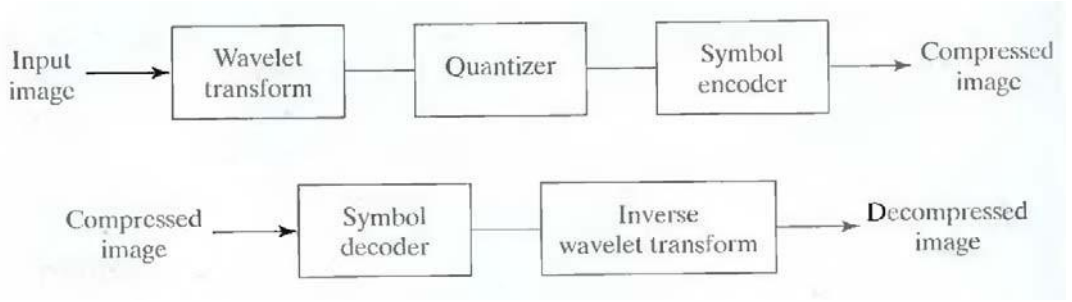


Fig.11 A wavelet coding system: (a) encoder; (b) decoder.

Since many of the computed coefficients carry little visual information, they can be quantized and coded to minimize intercoefficient and coding redundancy. Moreover, the quantization can be adapted to exploit any positional correlation across the P decomposition levels. One or more of the lossless coding methods, including run-length, Huffman, arithmetic, and bit-plane coding, can be incorporated into the final symbol coding step. Decoding is accomplished by inverting the encoding operations—with the exception of quantization, which cannot be reversed exactly.

The principal difference between the wavelet-based system and the