CHAPTER 1

Introduction to Image Processing

Of all of our inventions for mass communication, pictures still speak the most universally understood language.

–Walt Disney Company

LEARNING OBJECTIVES

This chapter provides an overview of digital image processing concepts. Image processing refers to the processing of visual information sources, such as images for some specific task, as per the application requirements. The objective of this chapter is to provide basic definitions that are associated with image processing and to give an overview of image types, imaging applications, and the digital image processing environment. The reader will become familiar with the following after studying the chapter:

- Elements of a digital image
- Types of images
- Fundamental steps in digital image processing
- Fundamental classes of image processing
- Survey of image processing applications

1.1 OVERVIEW OF IMAGE PROCESSING

Computers are faster and more accurate than human beings in processing numerical data. However, human beings score over computers in recognition capability. The human brain is so sophisticated that we recognize objects in a few seconds, without much difficulty. We may see a friend after ten years, yet recognize him/her in spite of the change in his/her appearance, as the method by which humans gather knowledge for recognition is very unique. Human beings use all the five sensory organs to gather knowledge about the outside world. Among these perceptions, visual information plays a major role in understanding the surroundings. Other kinds of sensory information are obtained from hearing, taste, smell, and touch. The old Chinese proverb 'A picture speaks a thousand words' rightly points out that images are very powerful tools in communication. With the advent of cheaper digital cameras and computer systems, we are witnessing a powerful

digital revolution, where images are being increasingly used to communicate ideas effectively.

We encounter images everywhere in our daily lives. We see many visual information sources such as paintings and photographs in magazines, journals, image galleries, digital libraries, newspapers, advertisement boards, television, and the Internet. Images are virtually everywhere! Many of us take digital snaps of important events in our lives and preserve them as digital albums. Then from the digital album, we print digital pictures and/or mail them to our friends to share our feelings of happiness and sorrow. However, images are not used merely for entertainment purposes. Doctors use medical images to diagnose problems for providing treatment. With modern technology, it is possible to image virtually all anatomical structures, which is of immense help to doctors in providing better treatment. Forensic imaging applications process fingerprints, faces, and irises to identify criminals. Industrial applications use imaging technology to count and analyse industrial components. Remote sensing applications use images sent by satellites to locate the minerals present in the earth. Thus, images find major applications in our everyday life.

Images are imitations of real-world objects. Often an image is a two-dimensional (2D) signal $f(x, y)$, where the values of the function $f(x, y)$ represent the amplitude or intensity of the image. For processing using digital computers, this image has to be converted into a discrete form using the process of sampling and quantization, known collectively as digitization. In image processing, the term 'image' is used to denote the image data that is sampled, quantized, and readily available in a form suitable for further processing by digital computers. Some authors use the term 'picture' to refer to analog or raw image data and the term 'image' to refer to digital data that is suitable for processing using digital computers. Broadly speaking, image processing is an area that deals with manipulation of visual information. One of the major objectives of image processing is to improve the quality of pictorial information for better human interpretation. Another objective is to facilitate the automatic machine interpretation of images.

.2 NATURE OF IMAGE PROCESSING

There are three scenarios or ways of acquiring an image—reflective mode imaging, emissive type imaging, and transmissive imaging. These are illustrated in Fig. 1.1. The radiation source shown in Fig. 1.1 is the light source. Can you imagine a world without light? Objects are perceived by the eye because of light. The sun, lamps, and clouds are all examples of radiation or light sources. The object is the target for which the image needs to be created. The object can be people, industrial components, or the anatomical structure of a patient. The objects can be two-dimensional, three-dimensional, or multidimensional mathematical functions involving many variables. For example, a printed document is a 2D object. Most real-world objects are 3D. *Reflective mode imaging* represents the simplest

form of imaging and uses a sensor to acquire the digital image. All video cameras, digital cameras, and scanners use some types of sensors for capturing the image. Image sensors are important components of imaging systems. They convert light energy to electric signals.

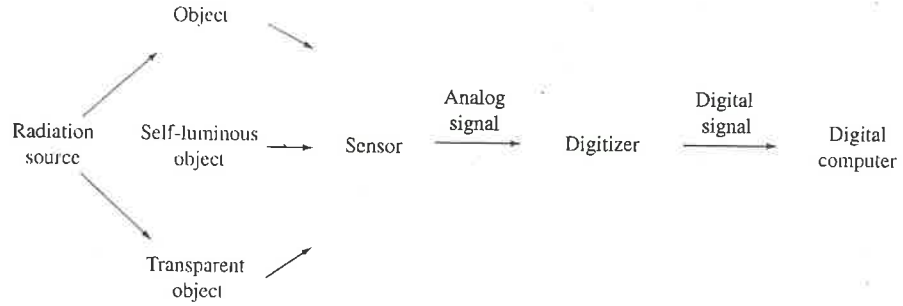


Fig. 1.1 Image processing environment

Emissive type imaging is the second type, where the images are acquired from self-luminous objects without the help of a radiation source. In emissive type imaging, the objects are self-luminous. The radiation emitted by the object is directly captured by the sensor to form an image. Thermal imaging is an example of emissive type imaging. In thermal imaging, a specialized thermal camera is used in low light situations to produce images of objects based on temperature. Other examples of emissive type imaging are magnetic resonance imaging (MRI) and positron emissive tomography (PET).

Transmissive imaging is the third type, where the radiation source illuminates the object. The absorption of radiation by the objects depends upon the nature of the material. Some of the radiation passes through the objects. The attenuated radiation is sensed into an image. This is called transmissive imaging. Examples of this kind of imaging are X-ray imaging, microscopic imaging, and ultrasound imaging.

The first major challenge in image processing is to acquire the image for further processing. Figure 1.1 shows three types of processing—optical, analog, and digital image processing. Optical image processing is the study of the radiation source, the object, and other optical processes involved. It refers to the processing of images using lenses and coherent light beams instead of computers. Human beings can see only the optical image. An optical image is the 2D projection of a 3D scene. This is a continuous distribution of light in a 2D surface and contains information about the object that is in focus. This is the kind of information that needs to be captured for the target image. *Optical image processing* is an area that deals with the object, optics, and how processes are applied to an image that is available in the form of reflected or transmitted light. The optical image is said to be available in optical form till it is converted into analog form.

An analog or continuous image is a continuous function $f(x, y)$, where x and y are two spatial coordinates. Analog signals are characterized by continuous signals varying with time. They are often referred to as pictures. The processes that are applied to the analog signal are called analog processes. *Analog image processing* is an area that deals with the processing of analog electrical signals using analog circuits. The imaging systems that use film for recording images are also known as analog imaging systems. In medical imaging, still films are used, as films provide better quality than digital systems.

The analog signal is often sampled, quantized, and converted into digital form using a digitizer. *Digitization* refers to the process of sampling and quantization. *Sampling* is the process of converting a continuous-valued image $f(x, y)$ into a discrete image, as computers cannot handle continuous data. So the main aim is to create a discretized version of the continuous data. Sampling is a reversible process, as it is possible to get the original image back. *Quantization* is the process of converting the sampled analog value of the function $f(x, y)$ into a discrete-valued integer. *Digital image processing* is an area that uses digital circuits, systems, and software algorithms to carry out the image processing operations. The image processing operations may include quality enhancement of an image, counting of objects, and image analysis.

Digital image processing has become very popular now, as digital images have many advantages over analog images. Some of the advantages are as follows:

1. It is easy to post-process the image. Small corrections can be made in the captured image using software.
2. It is easy to store the image in the digital memory.
3. It is possible to transmit the image over networks. So sharing an image is quite easy.
4. A digital image does not require any chemical process. So it is very environment friendly, as harmful film chemicals are not required or used.
5. It is easy to operate a digital camera.

The disadvantages of digital images are very few. Some of the disadvantages are the initial cost, problems associated with sensors such as high power consumption and potential equipment failure, and other security issues associated with the storage and transmission of digital images. Digital imaging is the technique practised now, as the advantages of digital image processing outweigh the disadvantages.

The final form of an image is the display image. The human eye can recognize only the optical form. So the digital image needs to be converted to optical form through the digital to analog conversion process.

1.3 IMAGE PROCESSING AND RELATED FIELDS

Image processing is an exciting interdisciplinary field that borrows ideas freely from many fields. Figure 1.2 illustrates the relationships between image processing and other related fields.

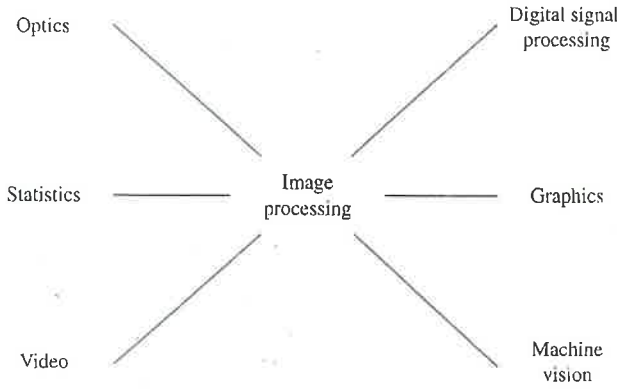


Fig. 1.2 Image processing and other closely related fields

1.3.1 Image Processing and Computer Graphics

Computer graphics and image processing are very closely related areas. Image processing deals with raster data or bitmaps, whereas computer graphics primarily deals with vector data. Raster data or bitmaps are stored in a 2D matrix form and often used to depict real images. However, vector images are composed of vectors, which represent the mathematical relationships between the objects. Vectors are lines or primitive curves that are used to describe an image. Vector graphics are often used to represent abstract, basic line drawings.

The algorithms in computer graphics often take numerical data as input and produce an image as output. However, in image processing, the input is often an image. The goal of image processing is to enhance the quality of the image to assist in interpreting it. Hence, the result of image processing is often an image or the description of an image. Thus, image processing is a logical extension of computer graphics and serves as a complementary field.

1.3.2 Image Processing and Signal Processing

Human beings interact with the environment by means of various signals. In digital signal processing, one often deals with the processing of a one-dimensional signal. In the domain of image processing, one deals with visual information that is often in two or more dimensions. Therefore, image processing is a logical extension of signal processing.

1.3.3 Image Processing and Machine Vision

The main goal of machine vision is to interpret the image and to extract its physical, geometric, or topological properties. Thus, the output of image processing operations can be subjected to more techniques, to produce additional information for interpretation.

Artificial vision is a vast field, with two main subfields—machine vision and computer vision. The domain of *machine vision* includes many aspects such as lighting and camera, as part of the implementation of industrial projects, since most of the applications associated with machine vision are automated visual inspection systems. The applications involving machine vision aim to inspect a large number of products and achieve improved quality controls. *Computer vision* is more ambitious. It tries to mimic the human visual system and is often associated with scene understanding. Most image processing algorithms produce results that can serve as the first input for machine vision algorithms.

1.3.4 Image Processing and Video Processing

Image processing is about still images. In fact, analog video cameras can be used to capture still images. A video can be considered as a collection of images indexed by time. Most image processing algorithms work with video readily. Thus, video processing is an extension of image processing. In addition, images are strongly related to multimedia, as the field of multimedia broadly includes the study of audio, video, images, graphics, and animation.

1.3.5 Image Processing and Optics

Optical image processing deals with lenses, light, lighting conditions, and associated optical circuits. The study of lenses and lighting conditions has an important role in the study of image processing.

1.3.6 Image Processing and Statistics

Image analysis is an area that concerns the extraction and analysis of object information from the image. Imaging applications involve both simple statistics such as counting and mensuration and complex statistics such as advanced statistical inference. So statistics play an important role in imaging applications. Image understanding is an area that applies statistical inferencing to extract more information from the image.

1.4 DIGITAL IMAGE REPRESENTATION

An image can be defined as a 2D signal that varies over the spatial coordinates x and y, and can be written mathematically as $f(x, y)$. Medical images such as magnetic resonance images and computerized tomography (CT) images are 3D images that can be represented as $f(x, y, z)$, where x, y, and z are spatial coordinates. A sample digital image and its matrix equivalent are shown in Figs 1.3(a) and 1.3(b).

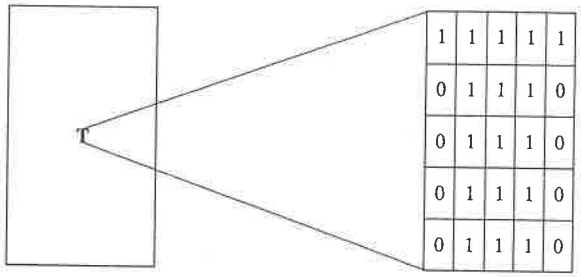


Fig. 1.3 Digital image representation (a) Small binary digital image (b) Equivalent image contents in matrix form

Figure 1.3(a) shows a displayed image. The source of the image is a matrix as shown in Fig. 1.3(b). The image has five rows and five columns. In general, the image can be written as a mathematical function $f(x, y)$ as follows:

$$f(x,y)=\begin{pmatrix} f(0,0) & f(0,1) & f(0,2) & \cdots & f(0,Y-1) \\ f(1,0) & f(1,1) & f(1,2) & \cdots & f(1,Y-1) \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ f(X-1,0) & f(X-1,1) & f(X-1,2) & \cdots & f(X-1,Y-1) \end{pmatrix}$$

In general, the image $f(x, y)$ is divided into X rows and Y columns. Thus, the coordinate ranges are $\{x=0, 1,\ldots, X-1\}$ and $\{y=0, 1, 2,\ldots, Y-1\}$. At the intersection of rows and columns, pixels are present. The word 'pixel' is an abbreviation of 'picture element'. The terms pixel, picture element, and pel are synonymous. A typical digital image consists of millions of pixels. Pixels are considered the building blocks of digital images, as they combine together to give a digital image. Pixels represent discrete data. Their meaning varies with context. A pixel can be considered as a single sensor, photosite (physical element of the sensor array of a digital camera), element of a matrix, or display element on a monitor.

The value of the function $f(x, y)$ at every point indexed by a row and a column is called *grey value* or *intensity* of the image. Generally, the value of the pixel is the intensity value of the image at that point. The intensity value is the sampled, quantized value of the light that is captured by the sensor at that point. It is a number and has no units. However, the value of the pixel is not always the intensity value. In an X-ray image, the value of the pixel indicates the attenuation of the X-ray at that point. In an MRI, the pixel value denotes the average MR signal intensity.

The number of rows in a digital image is called *vertical resolution*. The number of columns is called *horizontal resolution*. The number of rows and columns describes the dimensions of the image. The image size is often expressed in terms of the rectangular pixel dimensions of the array. Images can be of various sizes. Some examples of image

size are 256×256, 512×512, etc. For a digital camera, the image size is defined as the total number of pixels (specified in megapixels).

Resolution is an important characteristic of an imaging system. It is the ability of the imaging system to produce the smallest discernable details, that is the smallest sized object clearly, and differentiate it from the neighbouring small objects that are present in the image. Image resolution depends on two factors—optical resolution of the lens and spatial resolution. Optical resolution is covered in Chapter 2.

Spatial resolution of the image is very crucial as the digital image must show the object and its separation from the other spatial objects that are present in the image clearly and precisely. Consider a chart with vertical lines of width W. Let the space between the lines also be W. Then the line and the adjacent space constitute a line pair. The width of the line pair is $2W$, that is, W for the line and W for the space. Thus there are $1/2W$ line pairs per unit distance. A useful way to define resolution is the smallest number of line pairs per unit distance. The resolution can then be quantified as 200 line pairs per mm.

Spatial resolution depends on two parameters—the number of pixels of the image and the number of bits necessary for adequate intensity resolution, referred to as the bit depth. The numbers of pixels determine the quality of the digital image. The total number of pixels that are present in the digital image is the number of rows multiplied by the number of columns.

The choice of bit depth is very crucial and often depends on the precision of the measurement system. To represent the pixel intensity value, certain bits are required. For example, in binary images, the possible pixel values are 0 or 1. To represent two values, one bit is sufficient. The number of bits necessary to encode the pixel value is called *bit depth*. Bit depth is a power of two; it can be written as 2^m. In monochrome grey scale images (e.g., medical images such as X-rays and ultrasound images), the pixel values can be between 0 and 255. Hence, eight bits are used to represent the grey shades between 0 and 255 (as $2^8=256$). So the bit depth of grey scale images is 8. In colour images, the pixel value is characterized by both colour value and intensity value. So colour resolution refers to the number of bits used to represent the colour of the pixel. The set of all colours that can be represented by the bit depth is called *gamut* or *palette*.

So the total number of bits necessary to represent the image is

Number of rows × Number of columns × Bit depth

As discussed earlier, spatial resolution depends on the number of pixels present in the image and the bit depth. Keeping the number of pixels constant but reducing the quantization levels (bit depth) leads to a phenomenon called *false contouring*. On the other hand, the decrease in the number of pixels while retaining the quantization levels leads to a phenomenon called *checkerboard effect* (or *pixelization error*). These effects are discussed in detail in Chapter 2.

The concept of 2D images can be extended to 3D images also. A 3D image is a function $f(x, y, z)$, where x, y, and z are spatial coordinates. In 3D images, the term 'voxel' is used for pixel. Voxel is an abbreviation of 'volume element'.

1.5 TYPES OF IMAGES

There is no single accepted way of classifying images. They can be classified based on many criteria. Some ways in which images can be classified are shown in Fig. 1.4.

1.5.1 Based on Nature

Images can be broadly classified as natural and synthetic images. Natural images are, as the name implies, images of the natural objects obtained using devices such as cameras or scanners. Synthetic images are images that are generated using computer programs.

1.5.2 Based on Attributes

Based on attributes, images can be classified as raster images and vector graphics. Vector graphics use basic geometric attributes such as lines and circles, to describe an image. Hence the notion of resolution is practically not present in graphics. However, raster images (discussed in Section 1.3.1) are pixel-based. The quality of the raster images is dependent on the number of pixels. So operations such as enlarging or blowing-up of a raster image often result in quality reduction.

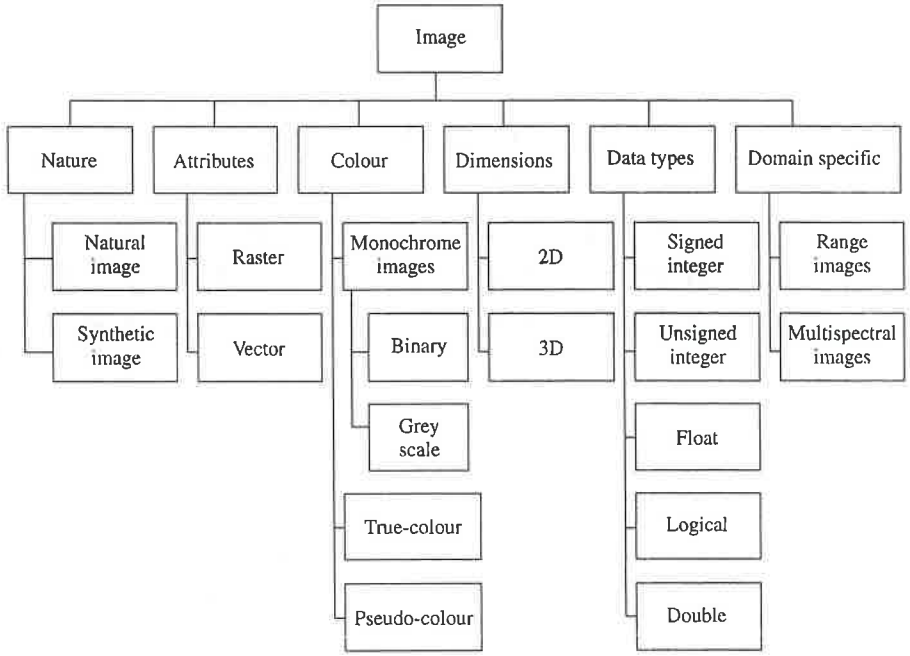


Fig. 1.4 Classification of images

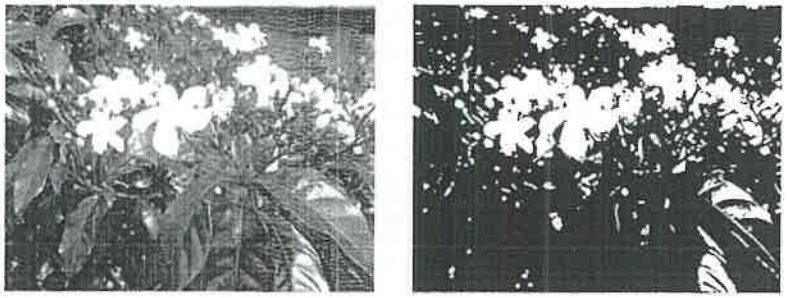
1.5.3 Based on Colour

Based on colour, images can be classified as grey scale, binary, true colour, and pseudocolour images. Grey scale and binary images are called monochrome images as there is no colour component in these images. True colour (or full colour) images represent the full range of available colours. So the images are almost similar to the actual object and hence called true colour images. In addition, true colour images do not use any lookup table but store the pixel information with full precision. On the other hand, pseudocolour images are false colour images where the colour is added artificially based on the interpretation of data.

1.5.3.1 Grey scale images

Grey scale images are different from binary images as they have many shades of grey between black and white. A sample grey scale image is shown in Fig. 1.5(a). These images are also called monochromatic as there is no colour component in the image, like in binary images. *Grey scale* is the term that refers to the range of shades between white and black or vice versa.

Eight bits ($2^8=256$) are enough to represent grey scale as the human visual system can distinguish only 32 different grey levels. The additional bits are necessary to cover noise margins. Most medical images such as X-rays, CT images, MRIs, and ultrasound images are grey scale images. These images may use more than eight bits. For example, CT images may require a range of 10–12 bits to accurately represent the image contrast.



(a) (b)

Fig. 1.5 Monochrome images (a) Grey scale image (b) Binary image

1.5.3.2 Binary images

In binary images, the pixels assume a value of 0 or 1. So one bit is sufficient to represent the pixel value. Binary images are also called bi-level images. In image processing, binary images are encountered in many ways.

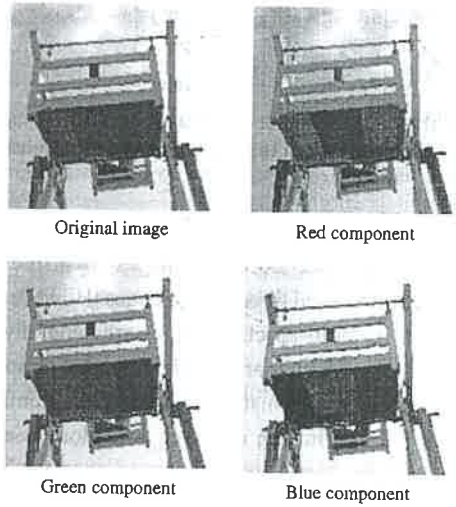
The binary image is created from a grey scale image using a threshold process. The pixel value is compared with the threshold value. If the pixel value of the grey scale image is greater than the threshold value, the pixel value in the binary image is considered as 1.

Otherwise, the pixel value is 0. The binary image created by applying the threshold process on the grey scale image in Fig. 1.5(a) is displayed in Fig. 1.5(b). It can be observed that most of the details are eliminated. However, binary images are often used in representing basic shapes and line drawings. They are also used as masks. In addition, image processing operations produce binary images at intermediate stages.

1.5.3.3 True colour images

In true colour images, the pixel has a colour that is obtained by mixing the primary colours red, green, and blue. Each colour component is represented like a grey scale image using eight bits. Mostly, true colour images use 24 bits to represent all the colours. Hence true colour images can be considered as three-band images. The number of colours that is possible is 256^3 (i.e., $256 \times 256 \times 256 = 1,67,77,216$ colours). Figure 1.6(a) shows a colour image and its three primary colour components. Figure 1.6(b) illustrates the general storage structure of the colour image. A display controller then uses a digital-to-analog converter (DAC) to convert the colour value to the pixel intensity of the monitor.

A special category of colour images is the indexed image. In most images, the full range of colours is not used. So it is better to reduce the number of bits by maintaining a colour map, gamut, or palette with the image. Figure 1.6(c) illustrates the storage structure of an indexed image. The pixel value can be considered as a pointer to the index, which contains the address of the colour map. The colour map has RGB components. Using this indexed approach, the number of bits required to represent the colours can be drastically reduced. The display controller uses a DAC to convert the RGB value to the pixel intensity of the monitor.



(a)

Fig. 1.6 True colour images (a) Original image and its colour components

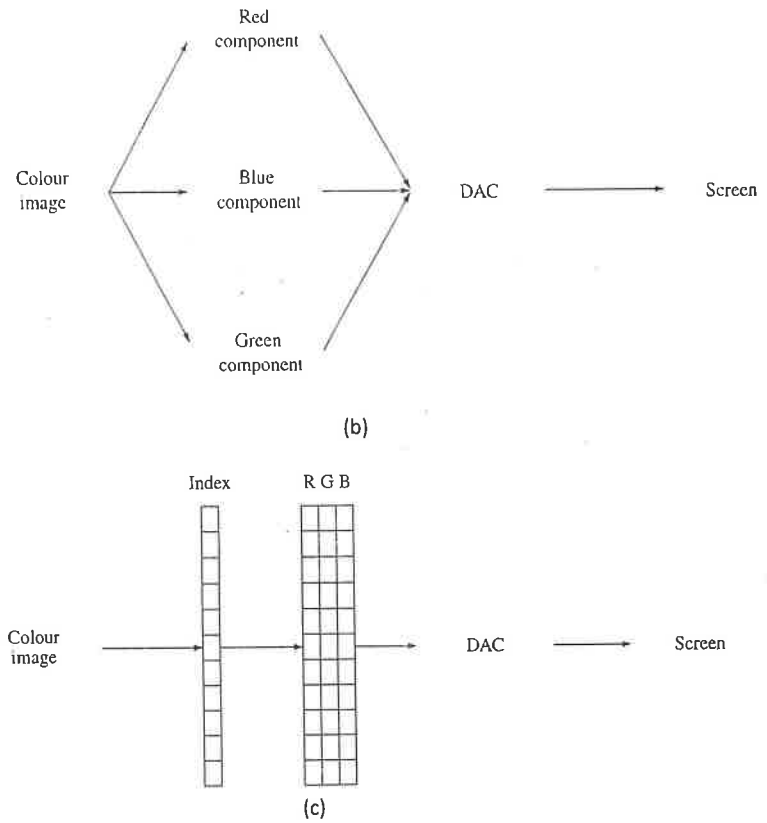


Fig. 1.6 (b) Storage structure of colour images (c) Storage structure of an indexed image [Refer to Oxford University Press (OUPI) website for colour images]

1.5.3.4 Pseudocolour images

Like true colour images, pseudocolour images are also used widely in image processing. True colour images are called three-band images. However, in remote sensing applications, multi-band images or multi-spectral images are generally used. These images, which are captured by satellites, contain many bands. A typical remote sensing image may have 3–11 bands in an image. This information is beyond the human perceptual range. Hence it is mostly not visible to the human observer. So colour is artificially added to these bands, so as to distinguish the bands and to increase operational convenience. These are called artificial colour or pseudocolour images. Pseudocolour images are popular in the medical domain also. For example, the Doppler colour image is a pseudocolour image.

Example 1.1 What is the storage requirement for a 1024×1024 binary image?

Solution For a binary image, one bit is sufficient for representing the pixel value. So the number of bits required will be 1024×1024×1 = 10,48,576 bits = 1,31,072 bytes = 131.072 Kb (Assume 1 Kb = 1000 bytes).

Example 1.2 What is the storage requirement for a 1024 × 1024 24-bit colour image?

Solution Since colour images are three-band images (red, green, and blue components), the storage requirement is 1024×1024×3 bytes = 31,45,728 bytes. If it is assumed that 1 Kb is 1000 bytes, the storage requirement is 3,145.728 Kb.

Example 1.3 A picture of physical size 2.5 inches by 2 inches is scanned at 150 dpi. How many pixels would be there in the image?

Solution The relation between the physical dimensions and the spatial resolution is simple. The pixel dimensions are obtained by multiplying the physical width and height by the scanned resolution. Therefore, the pixel dimension is as follows.

$$(2.5 \times 150) \times (2 \times 150)$$

$$= 375 \times 300 = 112500 \text{ pixels would be present}$$

Example 1.4 If a 375 × 300 grey-scale image needs to be sent across the channel of capacity 28 kbps, then how much transmission time is required?

Solution If the picture is grey scale, then 8 bits are used. Therefore, transmission time would be

$$= \frac{375 \times 300 \times 8}{28 \times 1000} = \frac{112500 \times 8}{28000} = 32.143 \text{ sec}$$

Example 1.5 Given a grey-scale image of size 5 inches by 6 inches scanned at the rate of 300 dpi, answer the following:

(a) How many bits are required to represent the image?
(b) How much time is required to transmit the image if the modem is 28 kbps?
(c) Repeat the aforementioned if it were a binary image.

Solution

(a) Number of bits required to represent grey-scale image (uses 8 bits)

$$= 5 \times 300 \times 6 \times 300 \times 8 = 1500 \times 1800 \times 8 = 21600000 \text{ bits}$$

(b) Total time taken to transmit image

$$= \frac{\text{Total number of bits in image}}{\text{Transmission Speed}} = \frac{21600000}{28000} = 771.43 \text{ sec}$$

(c) If it is binary image, then the number of bits required to represent binary image

$$= 5 \times 300 \times 6 \times 300 \times 1 = 1500 \times 1800 \times 1 = 2700000 \text{ bits}$$

The total transmission time would be $= \frac{\text{Total number of bits}}{\text{Transmission speed}} = \frac{2700000}{28000} = 96.429$ sec

1.5.4 Based on Dimensions

Images can be classified based on dimensions also. Normally, digital images are a 2D rectangular array of pixels. If another dimension, of depth or any other characteristic, is considered, it may be necessary to use a higher-order stack of images. A good example of a 3D image is a volume image, where pixels are called voxels. By '3D image', it is meant that the dimension of the target in the imaging system is 3D. The target of the imaging system may be a scene or an object. In medical imaging, some of the frequently encountered 3D images are CT images, MRIs, and microscopy images. Range images, which are often used in remote sensing applications, are also 3D images.

1.5.5 Based on Data Types

Images may be classified based on their data type. A binary image is a 1-bit image as one bit is sufficient to represent black and white pixels. Grey scale images are stored as one-byte (8-bit) or two-byte (16-bit) images. With one byte, it is possible to represent 2^8, that is 0–255 = 256 shades and with 16 bits, it is possible to represent 2^{16}, that is, 65,536 shades. Colour images often use 24 or 32 bits to represent the colour and intensity value.

Sometimes, image processing operations produce images with negative numbers, decimal fractions, and complex numbers. For example, Fourier transforms produce images involving complex numbers. To handle negative numbers, signed and unsigned integer types are used. In these data types, the first bit is used to encode whether the number is positive or negative. For example, the signed data type encodes the numbers from −128 to 127 where one bit is used to encode the sign. In general, an n-bit signed integer can represent integers from -2^{n-1} to $2^{n-1}-1$, a total of 2^n. Unsigned integers represent all integers from 0 to 2^n-1 with n bits.

Floating-point involves storing the data in scientific notation. For example, 1230 can be represented as 0.123×10^4, where 0.123 is called the significand and the power is called the exponent. There are many floating-point conventions.

The quality of such data representation is characterized by parameters such as data accuracy and precision. Data accuracy is the property of how well the pixel values of an image are able to represent the physical properties of the object that is being imaged. Data accuracy is an important parameter, as the failure to capture the actual physical properties of the image leads to the loss of vital information that can affect the quality of the application. While accuracy refers to the correctness of a measurement, precision refers to the repeatability of the measurement. In other words, repeated measurements of the physical properties of the object should give the same result. Most software use the data type 'double' to maintain precision as well as accuracy.

1.5.6 Domain Specific Images

Images can be classified based on the domains and applications where such images are encountered. The following are some of those images that are popular.

1.5.6.1. Range images

Range images are often encountered in computer vision. In range images, the pixel values denote the distance between the object and the camera. These images are also referred to as depth images. This is in contrast to all other images that have been discussed so far whose pixel values denote intensity and hence are often known as intensity images.

1.5.6.2. Multispectral images

Multispectral images are encountered mostly in remote sensing applications. These images are taken at different bands of visible or infrared regions of the electromagnetic wave. Just as a colour image is of three bands, multispectral images may have many bands that may include infrared and ultraviolet regions of the electromagnetic spectrum.

1.6 DIGITAL IMAGE PROCESSING OPERATIONS

The flow of an image processing operation is illustrated in Fig. 1.7. Image processing applications take an image as input and produce either an image or descriptions of the objects that are present in the image as output. Generally the output of image processing operations is another image. Brightness enhancement and contrast manipulation are examples of image processing operations.

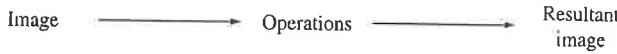


Fig. 1.7 Image processing operation

Figure 1.8 illustrates the general framework of an image analysis operation. The input for image analysis operations is in the form of an image. Image analysis operations produce a numerical output or descriptions of either the objects that are present in the image or the image itself. Some examples of image analysis operations are histogram of an image, and counting and gauging of objects.

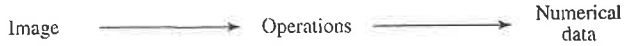


Fig. 1.8 Image analysis operation

Image processing and image analysis operations take the information from the image as it is. However, when the knowledge of the user is combined with image information, many intelligent applications can be designed. An image understanding operation is one such intelligent application that tries to mimic the human visual system. Image understanding applications aim to construct models using the knowledge constructs, and use the constructs to identify and analyse the spatial relations among the objects that are present in the image, just like the human visual system. Normally, image understanding applications are the primary goal of machine vision systems.

Generally, image processing operations are divided into two categories as follows:

1. Low-level operations 2. High-level operations

They are illustrated in Fig. 1.9.

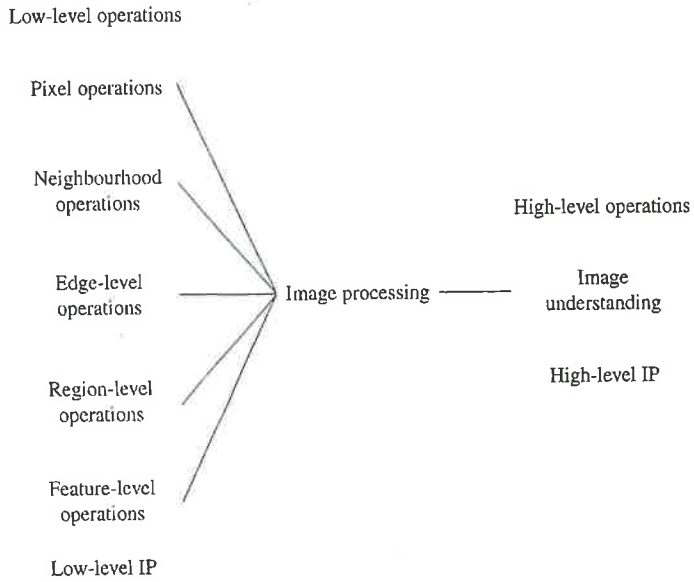


Fig. 1.9 Levels of image processing operations

Generally, low-level image processing is associated with traditional image processing, and there is a tendency to associate high-level operations with image understanding. There is no complete agreement among researchers as to what constitutes low-level processing. However, it is assumed that the knowledge requirements of low-level operations are very

less. In that context, operations such as image acquisition, preprocessing, and compression are considered as low-level operations.

Image segmentation and feature extraction deal with the extraction of necessary image portions and analysis of images for image features. Image features are essential information for recognition. Hence, segmentation and feature extraction are considered to be important areas, as these stages serve as a link between low-level and high-level image processing.

High-level image processing predominantly deals with image understanding. It deals with the interpretation of the image in a more meaningful manner, like an expert human observer. To do that, high-level operations construct models of the images, with knowledge constructs. High-level processing is based on knowledge, goals, and plans. Image understanding uses the concepts of artificial intelligence heavily to imitate human cognition. The process of image understanding involves the iteration of the following steps till adequate knowledge is gained:

1. Construction of the model of the real-world object or scene
2. Construction of the model from the image
3. A matching process, initiated between the real-world model and the model created from the image, which results in partial or complete matching
4. A feedback mechanism that invokes additional routines to update the models if necessary

This process is iteratively performed to create additional knowledge and feedback mechanisms. These steps are performed till the models converge to achieve the global goal. Naturally, these tasks are complex and often computationally intensive.

1.7 FUNDAMENTAL STEPS IN IMAGE PROCESSING

Automatic image interpretation is often desirable. A comparison of computer-based and manual interpretation is given in Table 1.1.

Table 1.1 Comparison of computer-based and manual interpretation

Computer-based interpretation	Manual interpretation
Computers are very accurate in performing numerical calculations, but less skilled in recognition compared to human beings.	Human beings are highly skilled in recognition, but slow in performing numerical calculations.
Computers are very fast.	Human beings are affected by many factors such as fatigue and boredom. Human errors are inevitable.
Computers are robust.	Human analysis is subjective. Often experts themselves differ from one another in interpretation. There are intra- and inter-operator differences.
Computers are flexible. They are easily configurable and easily deployable.	Human expertise is costly and less flexible.
Computer interpretation is reliable.	Human interpretation is subjective and variable. This affects reliability.

These points prove that computers can serve as effective tools in assisting experts. For example, in medical imaging applications, computer-based diagnosis may assist a physician by providing valuable information, so that the physician may use his expertise to arrive at the final diagnosis. This way, a computer-based diagnosis may lead to a more accurate diagnostic decision by a physician. The same logic applies to other domains as well.

Figure 1.10 illustrates the steps in automatic image analysis and interpretation. These steps are crucial for implementing any imaging application.

These steps are described as follows:

Image acquisition This step aims to obtain the digital image of the object.

Image enhancement This step aims to improve the quality of tne image so that the analysis of the images is reliable.

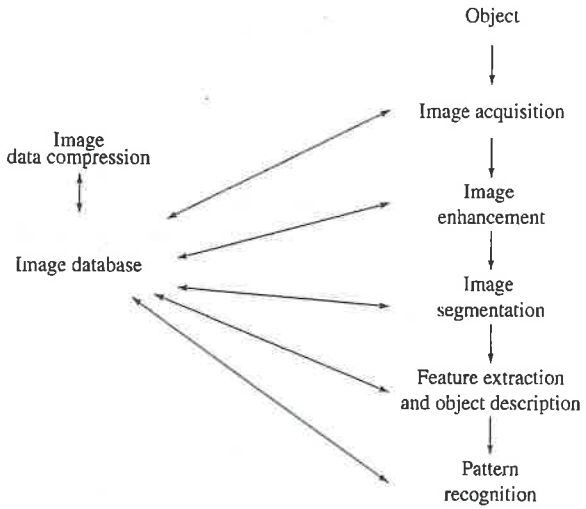


Fig. 1.10 Steps in image processing

Image segmentation This step divides the image into many sub-regions and extracts the regions that are necessary for further analysis. The portions of the image that are not necessary, such as image backgrounds (dictated by the imaging requirement), are discarded.

Feature extraction and object description Imaging applications use many routines for extraction of image features that are necessary for recognition. This is called image feature extraction step. The extracted object features are represented in meaningful data structures and the objects are described.

Pattern recognition This step is for identifying and recognizing the object that is present in the image, using the features generated in the earlier step and pattern recognition algorithms such as classification or clustering.

Image data compression and image database are the other important steps in image processing. Image databases are used to store the acquired images and the temporary images that are created during processing. The data compression step is crucial as it aims to reduce the storage requirement by removing the redundancies that are present in the image. This step is crucial as the storage requirements of images are very high.

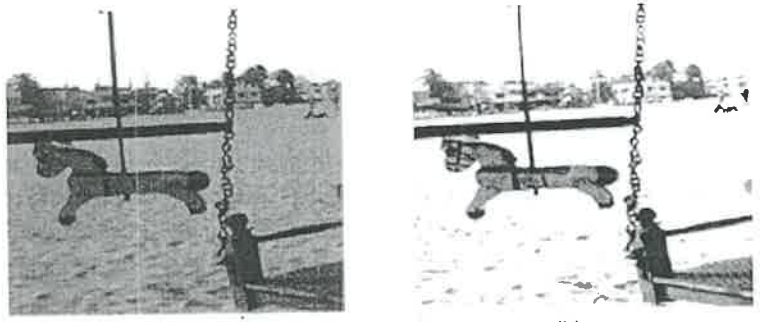
This basic model clearly serves as the baseline design for most imaging applications, where the aim is to recognize the object. Any practical application can be fitted into this baseline model. For example, the utility of this model can be explained for fingerprint recognition. The fingerprints are acquired in the image acquisition phase. Then the acquired fingerprint quality is enhanced for brightness and contrast. Then the portions that are necessary for fingerprint recognition such as loops, whorls, and minutiae are extracted in the image segmentation phase, represented as features, and analysed. Pattern recognition tasks such as classifiers can then use these features for recognizing the given fingerprint. The end result is the recognition of the fingerprint as either valid or invalid.

Amongst these stages, the following imaging operation classes are considered very fundamental. A class is a group of image operations that share the same objectives. The fundamental classes are as follows:

1. Image enhancement
2. Image restoration
3. Image compression
4. Image analysis
5. Image synthesis

1.7.1 Image Enhancement

Image enhancement is one of the most important classes of algorithms. Often, the captured image may not be of good quality (i.e., vital information that is necessary for the imaging application may not be available) because of factors such as noise, poor brightness, contrast, blur, or artefact. *Noise* is any unwanted signal. *Blur* is a disturbance that makes the image difficult for interpretation. *Artefacts* are features of the object that are not true. These are observational errors, including dust and scratches on the image surface, which complicate the process of accurate image interpretation. Therefore, it may be necessary to reduce the noise and to sharpen the details. These algorithms form the core of image enhancement. The dark image of Fig. 1.11(a) is enhanced by an image enhancement algorithm to increase the brightness of the image. A better quality image is shown in Fig. 1.11(b). The improvement can be assessed visually.

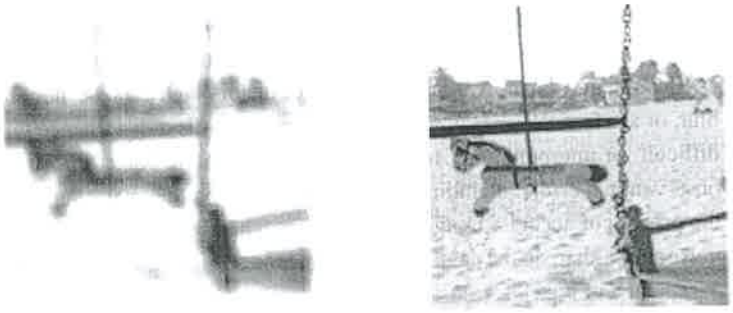


(a) (b)

Fig. 1.11 Image enhancement (a) Dark image (b) Enhanced image

1.7.2 Image Restoration

Image restoration is the objective way of improving the quality of the image. The goal of image restoration is the same as that of image enhancement. However, image restoration is different from image enhancement, as image restoration deals with degradations of extreme nature such as distortions created by the sensor system, poor lighting conditions, and artefacts. Image restoration is more mathematical and formal. Hence, image restoration problems are stated mathematically. Image restoration includes techniques such as inverse filtering and blind deconvolution algorithms. Sometimes, complete knowledge of the source of degradation is available. In that case, the simple inverse filtering process can be used to reverse the original degradations, as inverse filtering is the negation process for removing the degradations. However, if the causes of degradations are not known, then the degradations are estimated approximately and a process known as blind deconvolution is used to restore the original image. An example of motion blur and the restored image is shown in Figs 1.12(a) and 1.12(b).



(a) (b)

Fig. 1.12 Image restoration (a) Blurred image (b) Restored image

1.7.3 Image Compression

Multimedia objects occupy a lot of storage space. Often, these images need to be transmitted across a channel to a remote imaging system in imaging applications such as telemedicine. Hence in such a case, the storage and transmission of the image becomes an important task. Image compression algorithms reduce the data that is needed to describe the object, by eliminating the redundancies that are present in the image.

There are two classes of image compression algorithms. One class is lossless compression algorithms and the other is lossy compression algorithms. Lossless compression algorithms preserve the information that is very critical, and are useful in medical domains where even a subtle feature may contain valuable information. Lossy compression algorithms are used where the loss of image data cannot be perceived by the human observer or the loss of information is acceptable. Figure 1.13(a) illustrates an original image and the application of lossy compression on it to yield the resultant images shown in Figs 1.13(b) and 1.13(c) with a quality of 95% and 5%, respectively. It can be observed that loss of quality is not perceivable by the human eye. This illustrates that the image has more redundant data. Many imaging and video applications such as video conferencing require lossy compression algorithms to manage the size of the multimedia objects.

(a) (b) (c)

Fig. 1.13 Image compression (a) Original image (b) Image quality at 95% (c) Image quality at 5%

1.7.4 Image Analysis

Often, machine vision systems require image measurement. This includes measurement of shape, size, texture, and colour of the objects that are present in the image. Hence, image analysis is a very important class of algorithms that takes images as input and produces numerical and graphical information based on the characteristics of the image data. Image analysis comprises, but is not limited to, classification of the objects, performing statistical tasks, and providing extraction and description of the scene for ultimate interpretation. One example of image analysis is plotting the histogram of an image.

Figures 1.14(a) and 1.14(b) illustrate an image and its histogram. Histogram is a simple image analysis technique. It illustrates the distribution of grey levels of an image in the form of a table or graph. Based on the histogram, one can obtain information about the quality of the image. The darkness of the image is manifested in the histogram, as the dynamic ranges of the pixels are not good. Perhaps the presence of all pixels as a cluster illustrates that the image needs to be improved. In addition, image analysis involves finding measurements of the objects such as mean and variance. Some of the statistical measurements are shown in Fig. 1.14(b).

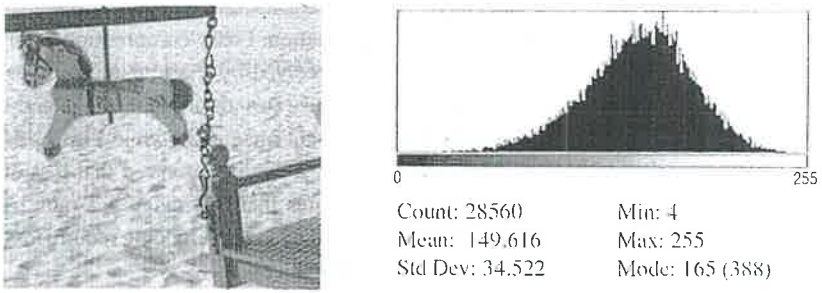


Fig. 1.14 Image analysis (a) Original image (b) Histogram and its statistics

1.7.5 Image Synthesis

Image synthesis deals with the creation of images from other images or non-image data. Image synthesis is used to create images that are not available physically or cannot be acquired using any imaging procedure. The medical imaging domain uses image synthesis extensively. A CT is a reconstructed image. For example, the image shown in Fig. 1.15 is a simulated image as it is neither scanned nor captured, but created through software. These simulated images are useful for presentation and experimental purposes as benchmark and test images.

Fig. 1.15 Sample synthetic grating

1.8 IMAGE PROCESSING APPLICATIONS

This section briefly surveys image processing applications. The survey of imaging applications can be from the viewpoint of either the electromagnetic spectrum or the application domain.

1.8.1 Survey of Image Processing Applications Based on EM Radiation

The survey from the electromagnetic domain is justified as visible light is part of the electromagnetic spectrum. Electromagnetic waves are waves of energy that have both electric and magnetic characteristics and which can travel at the speed of light in vacuum. Electromagnetic waves can be classified based on the frequency or wavelength and its select parts are shown in Table 1.2.

Table 1.2 Select parts of the electromagnetic spectrum

Types of radiation	Frequency range (in Hertz)	Wave length (in cm)	Nature of imaging and its relevance for image processing
Radio waves	10^5–10^{10}	$> 10^9$	AM/FM radio
Microwave	10^{10}–10^{12}	10^9–10^6	Radar imaging
Infrared	10^{12}–10^{14}	10^3–7000	Thermal imaging
Visible light	4–7.5 × 10^{14}	7000–4000	Optical
Ultraviolet	10^{15}–10^{17}	4000–10	Optical
X-rays	10^{17}–10^{20}	10–0.1	Medical and industrial
Gamma rays	10^{20}–10^{24}	< 0.1	Medical

Image processing applications fully exploit the entire range of the electromagnetic spectrum and some of the applications are given here.

Radio waves Radio waves have the lowest energy level and are used in radio and video broadcast as well as by mobile phones. Radio waves are used extensively in radios, satellites, radar, and computer networks for transmission of data. In image processing domain, radio waves are useful in two areas—remote sensing and medical imaging. Hydrogen gas in space releases energy in low frequency and is collected as radio waves. In radar systems, the released energy is collected back, which helps to image the surface of the earth and is useful in predicting weather forecasts. In medical imaging, radio waves are used in the form of magnetic resonance imaging (MRI). A patient is placed on a powerful magnet, and then radio waves are passed through the patient's body in short pulses. The responding pulses of the radio waves are collected by the computer, which determines the location of these signals and their strength. MRI is capable of detecting soft tissues, and image processing applications can be used for analysing MRI images.

Microwaves Microwaves are useful in radar and medical imaging. Radar is an abbreviation of RAdio Detection And Ranging. Radar systems use microwaves in addition to radio waves at higher frequency and are used in applications such as weather forecasting, imaging planetary surfaces, and determining earth resources. Synthetic aperture radar (SAR) is a tool for detecting targets and its velocity. The concept of microwaves is used in medical imaging to probe the magnetic resonance (MR) properties of electrons. This phenomenon

CHAPTER 3

Digital Image Processing Operations

Photography is the simplest thing in the world, but it is incredibly complicated to make it really work.

–Martin Parr

LEARNING OBJECTIVES

This chapter provides an overview of image processing operations. The concepts of image topology such as neighbourhood and connectivity are introduced in this chapter. The important arithmetic, logical, statistical, spatial, geometrical, and set operations are discussed. After studying this chapter, the reader will become familiar with the following:

- Concepts of image geometry and topology
- Image processing operations
- Logical operations
- Geometrical operations
- Interpolation techniques
- Set operations
- Spatial operations

3.1 BASIC RELATIONSHIPS AND DISTANCE METRICS

This chapter discusses the image processing operations. To carry out any image processing operation, the image must be stored in the memory in a suitable form. Chapter 2 discusses how the images are sampled and quantized with rectangular grids. During this sampling process, the digital image is a set of discrete pixels arranged in a rectangular grid. The pixel itself assumes the shape of a small rectangle. If the width and height of the pixel are the same, the shape of the pixel is a square. Even though grids of other shapes such as hexagonal and triangular are available, rectangular grids are widely used as they are intuitive, simple, and are inherently used in the discretization of images by CCD (charge coupled device) cameras and scanners.

3.1.1 Image Coordinate System

Images can be easily represented as a two-dimensional array or matrix. The popularity of the matrix form is due to the fact that most of the programming languages support 2D array data structure and can easily implement matrix-level computation.

Pixels can be visualized logically and physically. Logical pixels specify the points of a continuous 2D function. These are logical in the sense that they specify a location but occupy no physical area. Normally, this is represented in the Cartesian first coordinate system. Physical pixels, on the other hand, occupy a small amount of space when displayed on the output device. Digitized images indicating physical pixels are represented in the Cartesian fourth coordinate system. For example, an analog image of size 3 × 3 is represented in the first quadrant of the Cartesian coordinate system as shown in Fig. 3.1.

2	$f(0,2)$	$f(1,2)$	$f(2,2)$
1	$f(0,1)$	$f(1,1)$	$f(2,1)$
$y=0$	$f(0,0)$	$f(1,0)$	$f(2,0)$
	$x=0$	1	2

Fig. 3.1 Analog image $f(x, y)$ in the first quadrant of Cartesian coordinate system

Figure 3.1 illustrates an image $f(x, y)$ of dimension 3 × 3, where $f(0, 0)$ is the bottom left corner. Since it starts from the coordinate position (0, 0), it ends with $f(2, 2)$, that is, $x = 0, 1, 2, \ldots, M-1$ and $y = 0, 1, 2, \ldots, N-1$. x and y define the dimensions of the image.

However, in digital image processing, the discrete form of the image is often used. Discrete images are usually represented in the fourth quadrant of the Cartesian coordinate system. A discrete image $f(x, y)$, of dimension 3 × 3, is shown in Fig. 3.2(a).

Many programming environments including MATLAB start with an index of (1, 1). The equivalent representation of the given matrix is shown in Fig. 3.2(b).

The coordinates used for discrete image is, by default, the fourth quadrant of the Cartesian system.

	$y=0$	$y=1$	$y=2$
$x=0$	$f(0,0)$	$f(0,1)$	$f(0,2)$
$x=1$	$f(1,0)$	$f(1,1)$	$f(1,2)$
$x=2$	$f(2,0)$	$f(2,1)$	$f(2,2)$

(a)

	$y=1$	$y=2$	$y=3$
$x=1$	$f(1,1)$	$f(1,2)$	$f(1,3)$
$x=2$	$f(2,1)$	$f(2,2)$	$f(2,3)$
$x=3$	$f(3,1)$	$f(3,2)$	$f(3,3)$

(b)

Fig. 3.2 Discrete image (a) Image in the fourth quadrant of Cartesian coordinate system (b) Image coordinates as handled by software environments such as MATLAB

3.1.2 Image Topology

Image topology is a branch of image processing that deals with the fundamental properties of the image such as image neighbourhood, paths among pixels, boundary, and connected components. It characterizes the image with topological properties such as neighbourhood, adjacency, and connectivity. *Neighbourhood* is fundamental to understanding image topology. In the simplest case, the neighbours of a given reference pixel are those pixels with which the given reference pixel shares its edges and corners.

$$\begin{pmatrix} 0 & X & 0 \\ X & p(x,y) & X \\ 0 & X & 0 \end{pmatrix}$$

Fig. 3.3 4-Neighbourhood $N_4(p)$

In $N_4(p)$, the reference pixel $p(x, y)$ at the coordinate position (x, y) has two horizontal and two vertical pixels as neighbours. This is shown graphically in Fig. 3.3.

The set of pixels $\{(x+1, y), (x-1, y), (x, y+1), (x, y-1)\}$, called the 4-neighbours of p, is denoted as $N_4(p)$. Thus 4-neighbourhood includes the four direct neighbours of the pixel $p(x, y)$. By duality, these are pixels that share a common edge with the given reference pixel $p(x,y)$. Similarly, the pixel may have four diagonal neighbours. They are $(x-1, y-1)$, $(x+1, y+1)$, $(x-1, y+1)$, and $(x+1, y-1)$. The diagonal pixels for the reference pixel $p(x, y)$ are shown graphically in Fig. 3.4.

$$\begin{pmatrix} X & 0 & X \\ 0 & p(x,y) & 0 \\ X & 0 & X \end{pmatrix}$$

Fig. 3.4 Diagonal elements $N_D(p)$

The diagonal neighbours of pixel $p(x, y)$ are represented as $N_D(p)$. The 4-neighbourhood and N_D are collectively called the 8-neighbourhood. This refers to all the neighbours and pixels that share a common corner with the reference pixel $p(x, y)$. These pixels are called indirect neighbours. This is represented as $N_8(p)$ and is shown graphically in Fig. 3.5. The set of pixels $N_8(x) = N_D(x) \cup N_4(x)$.

$$\begin{pmatrix} X & X & X \\ X & p(x,y) & X \\ X & X & X \end{pmatrix}$$

Fig. 3.5 8-Neighbourhood $N_8(p)$

3.1.3 Connectivity

The relationship between two or more pixels is defined by pixel connectivity. Connectivity information is used to establish the boundaries of the objects. The pixels p and q are said to be connected if certain conditions on pixel brightness specified by the set V and spatial adjacency are satisfied. For a binary image, this set V will be $\{0, 1\}$ and for grey scale images, V might be any range of grey levels.

4-Connectivity The pixels p and q are said to be in 4-connectivity when both have the same values as specified by the set V and if q is said to be in the set $N_4(p)$. This implies any path from p to q on which every other pixel is 4-connected to the next pixel.

8-Connectivity It is assumed that the pixels p and q share a common grey scale value. The pixels p and q are said to be in 8-connectivity if q is in the set $N_8(p)$.

Mixed connectivity Mixed connectivity is also known as m-connectivity. Two pixels p and q are said to be in m-connectivity when

1. q is in $N_4(p)$ or
2. q is in $N_D(p)$ and the intersection of $N_4(p)$ and $N_4(q)$ is empty.

For example, Fig. 3.6 shows 8-connectivity when $V = \{0, 1\}$.

8-Connectivity is shown as lines. Here, a multiple path or loop is present. In m-connectivity, there are no such multiple paths. The m-connectivity for the image in Fig. 3.6 is as shown in Fig. 3.7.

It can be observed that the multiple paths have been removed.

3.1.4 Relations

A binary relation between two pixels a and b, denoted as aRb, specifies a pair of elements of an image. For example, consider the image pattern given in Fig. 3.8.

The set is given as $A = \{x_1, x_2, x_3\}$. The set based on the 4-connectivity relation is given as $A = \{x_1, x_2\}$. It can be observed that $x3$ is ignored as it is not connected to any other element of the image by 4-connectivity.

The following are the properties of the binary relations:

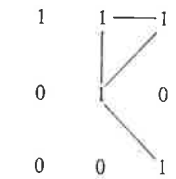


Fig. 3.6 8-Connectivity represented as lines

Reflexive For any element a in the set A, if the relation aRa holds, this is known as a reflexive relation.

Symmetric If aRb implies that bRa also exists, this is known as a symmetric relation.

Transitive If the relations aRb and bRc exist, it implies that the relationship aRc also exists. This is called the transitivity property.

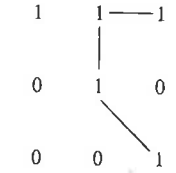


Fig. 3.7 m-Connectivity

If all these three properties hold, the relationship is called an *equivalence relation*. The relationships can be expressed as a matrix. Assume that the set A is divided into K disjoint sets, where K is an integer ranging from 1 to ∞. If a relation aRb exists, and if a and b belong to the same set, the relationship is said to be an equivalence relation. Transitive closure implies that if aRb and bRc exist, aRc also exists. The set containing all the implied relations is called the transitive closure of R, and is denoted as $R+$.

Fig. 3.8 Image pattern

3.1.5 Distance Measures

The distance between the pixels p and q in an image can be given by distance measures such as Euclidian distance, D_4 distance, and D_8 distance. Consider three pixels p, q, and z. If the coordinates of the pixels are $P(x, y)$, $Q(s, t)$, and $Z(u, w)$ as shown in Fig. 3.9, the distances between the pixels can be calculated.

The distance function can be called metric if the following properties are satisfied:

1. $D(p, q)$ is well-defined and finite for all p and q.
2. $D(p, q) \geq 0$ if $p = q$, then $D(p, q) = 0$.
3. The distance $D(p, q) = D(q, p)$.
4. $D(p, q) + D(q, z) \geq D(p, z)$. This is called the property of triangular inequality.

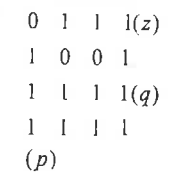


Fig. 3.9 Sample image

The Euclidean distance between the pixels p and q, with coordinates (x, y) and (s, t), respectively, can be defined as

$$D_e(p,q) = \sqrt{(x-s)^2 + (y-t)^2}$$

The advantage of the Euclidean distance is its simplicity. However, since its calculation involves a square root operation, it is computationally very costly.

The D_4 distance or city block distance can be simply calculated as

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

The D_8 distance or chessboard distance can be calculated as

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

Example 3.1 Let $V = \{0, 1\}$. Compute the D_e, D_4, D_8, and D_m distances between two pixels p and q. Let the pixel coordinates of p and q be (3, 0) and (2, 3), respectively, for the image shown in Fig. 3.10.

Find the distance measures.

	0	1	2	3
0	0	1	1	1
1	1	0	0	1
2	1	1	1	1(q)
3	1	1	1	1

(p)

Fig. 3.10 Sample image

Solution The Euclidean distance is

$$D_e = \sqrt{(x-s)^2 + (y-t)^2} = \sqrt{(3-2)^2 + (0-3)^2}$$

$$= \sqrt{1+9} = \sqrt{10}$$

$$D_4 = |x-s| + |y-t| = |3-2| + |0-3|$$

$$= 1 + 3 = 4$$

$$D_8 = \max\left(|x-s|, |y-t|\right) = \max\left(|3-2|, |0-3|\right)$$

$$= \max(1, 3) = 3$$

The distances can be checked with Figs 3.11(a)–3.11(d). The transition from one element to another is a hop. It can be verified that the calculations match with the hops in Fig. 3.11. The distance D_m depends on the values of the set V. The implication of the set V is that the path should be constructed only using the elements of V. So if the values of the set are changed, the path also changes.

The simplest D_m distance can be calculated along the diagonal path. The distance is 3. Suppose the set $V = \{1\}$, the path of distance D_m also changes. It is shown in Fig. 3.11.

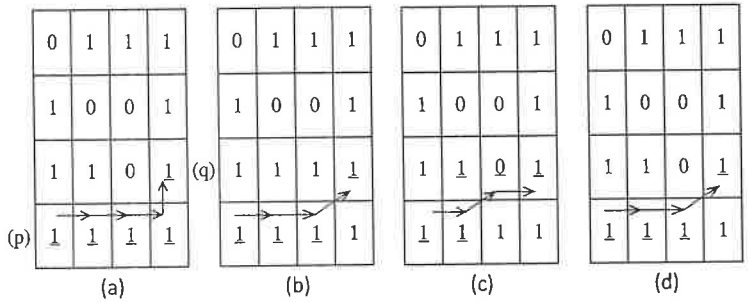


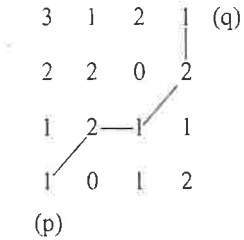
Fig. 3.11 Distance measures (a) Distance D_1 (b) Distance D_0 when $V = \{0, 1\}$ (c) Distance D_m when $V = \{0, 1\}$ (d) Distance D_m when $V = \{1\}$

Example 3.2 Consider the following image with marked pixels p and q.

3 1 2 1 (q)
2 2 0 2
1 2 1 1
1 0 1 2
(p)

If $V = \{1, 2\}$, find the shortest m-path between pixels p and q.

Solution The shortest path for set V {1,2} is shown here:



The shortest m-path is 1 – 2 – 1 – 2 –1. Therefore, D_m distance is 4.

3.1.6 Important Image Characteristics

Some important characteristics of images are as follows:

1. The set of pixels that has connectivity in a binary image is said to be characterized by the connected set.
2. A digital path or curve from pixel p to another pixel q is a set of points $p_1, p_2, \ldots, p_n$. If the coordinates of those points are (x_0, y_0), (x_1, y_1), …, (x_n, y_n), then $p = (x_0, y_0)$ and $q = (x_n, y_n)$. The number of pixels is called the length. If $x_0 = x_n$ and $y_0 = y_n$, then the path is called a closed path.
3. R is called a region if it is a connected component.
4. If a path between any two pixels p and q lies within the connected set S, it is called a connected component of S. If the set has only one connected component, then the set S is called a connected set. A connected set is called a region.
5. Two regions R_1 and R_2 are called adjacent if the union of these sets also forms a connected component. If the regions are not adjacent, it is called disjoint set. In Fig. 3.12, two regions R_1 and R_2 are shown. These regions are 8-connected because the pixels (underlined pixel '1') have 8-connectivity. If the regions are not adjacent, they are called disjoint.
6. The border of the image is called contour or boundary. A boundary is a set of pixels covering a region that has one or more neighbours outside the region. Typically, in a binary image, there is a foreground object and a background object. The border of the foreground object may have at least one neighbour in the background. If the border pixels are within the region itself, it is called inner boundary. This need not be closed.

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

Region 1 Region 2

Fig. 3.12 Neighbouring regions

7. Edges are present wherever there is an abrupt intensity change among the pixels. Edges are similar to boundaries, but may or may not be connected. If edges are disjoint, they have to be linked together by edge linking algorithms. However, boundaries are global and have a closed path. Figure 3.13 illustrates two regions and an edge. It can be observed that edges provide an outline of the object. The pixels that are covered by the edges lead to regions.

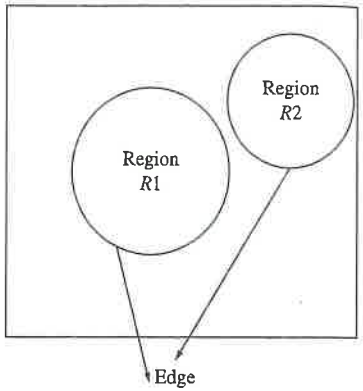


Fig. 3.13 Edge and regions

3.2 CLASSIFICATION OF IMAGE PROCESSING OPERATIONS

There are various ways to classify image operations. The reason for categorizing the operations is to gain an insight into the nature of the operations, the expected results, and the kind of computational burden that is associated with them.

One way of categorizing the operations based on neighbourhood is as follows:

1. Point operations 2. Local operations 3. Global operations

Point operations are those whose output value at a specific coordinate is dependent only on the input value. A local operation is one whose output value at a specific coordinate is dependent on the input values in the neighbourhood of that pixel. Global operations are those whose output value at a specific coordinate is dependent on all the values in the input image.

Another way of categorizing operations is as follows:

1. Linear operations 2. Non-linear operations

An operator is called a linear operator if it obeys the following rules of additivity and homogeneity.

1. Property of additivity

$$H(a_1 f_1(x,y) + a_2 f_2(x,y)) = H(a_1 f_1(x,y)) + H(a_2 f_2(x,y))$$
$$= a_1 H(f_1(x,y)) + a_2 H(f_2(x,y))$$
$$= a_1 \times g_1(x,y) + a_2 \times g_2(x,y)$$

2. Property of homogeneity

$$H(k f_1(x,y)) = kH(f_1(x,y)) = kg_1(x,y)$$

A non-linear operator, as the name suggests, does not follow these rules.

Image operations are array operations. These operations are done on a pixel-by-pixel basis. Array operations are different from matrix operations. For example, consider two images

$$F_1 = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \text{ and } F_2 = \begin{pmatrix} E & F \\ G & H \end{pmatrix}$$

The multiplication of F_1 and F_2 is element-wise, as follows:

$$F_1 \times F_2 = \begin{pmatrix} AE & BF \\ CG & HD \end{pmatrix}$$

In addition, one can observe that $F_1 \times F_2 = F_2 \times F_1$, whereas matrix multiplication is clearly different, since in matrices, $A \times B \neq B \times A$. By default, image operations are array operations only.

3.2.1 Arithmetic Operations

Arithmetic operations include image addition, subtraction, multiplication, division, and blending. The following sections discuss the usage of these operations.

3.2.1.1 Image addition

Two images can be added in a direct manner, as given by

$$g(x,y) = f_1(x,y) + f_2(x,y)$$

The pixels of the input images $f_1(x,y)$ and $f_2(x,y)$ are added to obtain the resultant image $g(x,y)$. Figure 3.14 shows the effect of adding a noise pattern to an image. However, during the image addition process, care should be taken to ensure that the sum does not cross the allowed range. For example, in a grey scale image, the allowed range is 0–255, using eight bits. If the sum is above the allowed range, the pixel value is set to the maximum allowed value. The range values of select MATLAB data types are listed in Table 3.1.

Table 3.1 Data type and allowed ranges

S. no.	Data type	Data range
1	Uint8	0–255
2	Uint16	0–65,535
3	Uint32	0–4,29,49,67,295
4	Uint64	0–1,84,46,74,40,73,70,95,51,615

Similarly, it is possible to add a constant value to a single image, as follows:

$$g(x,y) = f_1(x,y) + k$$

If the value of k is larger than 0, the overall brightness is increased. Figure 3.14(d) illustrates that the addition of the constant 50 increases the brightness of the image. Why?

The brightness of an image is the average pixel intensity of an image. If a positive or negative constant is added to all the pixels of an image, the average pixel intensity of the image increases or decreases, respectively. This is covered in Section 3.2.1.2. The practical applications of image addition are as follows:

1. To create double exposure. Double exposure is the technique of superimposing an image on another image to produce the resultant. This gives a scenario equivalent to exposing a film to two pictures. This is illustrated in Figs 3.14(a)–3.14(c).
2. To increase the brightness of an image.

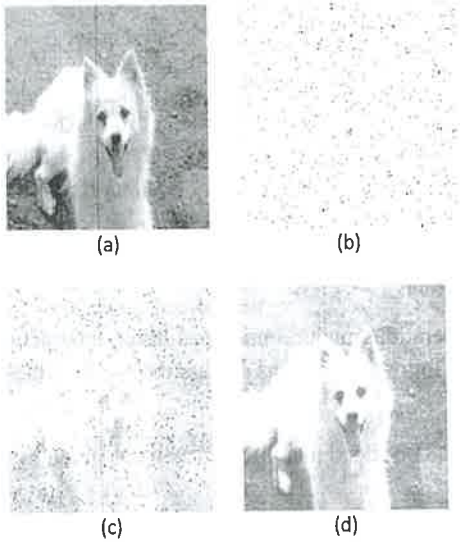


Fig. 3.14 Results of the image addition operation (a) Image 1 (b) Image 2 (c) Addition of images 1 and 2 (d) Addition of image 1 and constant 50

3.2.1.2 *Image subtraction*

The subtraction of two images can be done as follows. Consider

$$g(x, y) = f_1(x, y) - f_2(x, y)$$

where $f_1(x, y)$ and $f_2(x, y)$ are two input images and $g(x, y)$ is the output image. To avoid negative values, it is desirable to find the modulus of the difference as

$$g(x, y) = |f_1(x, y) - f_2(x, y)|$$

It is also possible to subtract a constant value k from the image, i.e., $g(x, y) = |f_1(x, y) - \mathrm{k}|$, as k is constant. As discussed earlier, the decrease in the average intensity reduces the brightness of the image. Some of the practical applications of image subtraction are as follows:

1. Background elimination
2. Brightness reduction
3. Change detection

If there is no difference between the frames, the subtraction process yields zero, and if there is any difference, it indicates the change. Figures 3.15(a)–3.15(d) show the difference between the images. In addition, it illustrates that the subtraction of a constant results in a decrease of the brightness.

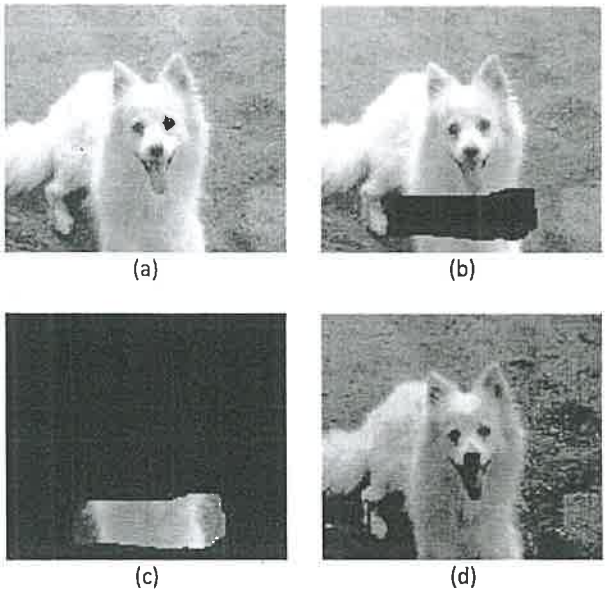


Fig. 3.15 Results of the image subtraction operation (a) Image 1 (b) Image 2 (c) Subtraction of images 1 and 2 (d) Subtraction of constant 50 from image 1

3.2.1.3 *Image multiplication*

Image multiplication can be done in the following manner:
Consider

$$g(x, y) = f_1(x, y) \times f_2(x, y)$$

Here $f_1(x, y)$ and $f_2(x, y)$ are two input images and $g(x, y)$ is the output image. If the multiplied value crosses the maximum value of the data type of the images, the value of the pixel is reset to the maximum allowed value. Similarly, scaling by a constant can be performed as

$$g(x, y) = f(x, y) \times k$$

where k is a constant.

If k is greater than 1, the overall contrast increases. If k is less than 1, the contrast decreases. The brightness and contrast can be manipulated together as

$$g(x, y) = af(x, y) + k$$

Here, the parameters a and k are used to manipulate the brightness and contrast of the input image. $g(x, y)$ is the output image. Some of the practical applications of image multiplication are as follows:

1. It increases contrast. If a fraction less than 1 is multiplied with the image, it results in decrease of contrast. Figure 3.16 shows that by multiplying a factor of 1.25 with the original image, the contrast of the image increases.
2. It is useful for designing filter masks.
3. It is useful for creating a mask to highlight the area of interest.

Fig. 3.16 Result of multiplication operation (image × 1.25) resulting in good contrast

3.2.1.4 Image division

Similar to the other operations, division can be performed as

$$g(x, y) = \frac{f_1(x, y)}{f_2(x, y)}$$

where $f_1(x, y)$ and $f_2(x, y)$ are two input images and $g(x, y)$ is the output image.

The division process may result in floating-point numbers. Hence, the float data type should be used in programming. Improper data type specification of the image may result in loss of information. Division using a constant can also be performed as

$$g(x, y) = \frac{f(x, y)}{k}, \text{ where } k \text{ is a constant.}$$

Some of the practical applications of image division are as follows:

1. Change detection
2. Separation of luminance and reflectance components
3. Contrast reduction. Figure 3.17(a) shows such an effect when the original image is divided by 1.25.

Figures 3.17(b)–3.17(e) show the multiplication and division operations used to create a mask. It can be observed that image 2 is used as a mask. The multiplication of image 1 with image 2 results in highlighting certain portions of image 1 while suppressing the other portions. It can be observed that division yields back the original image.

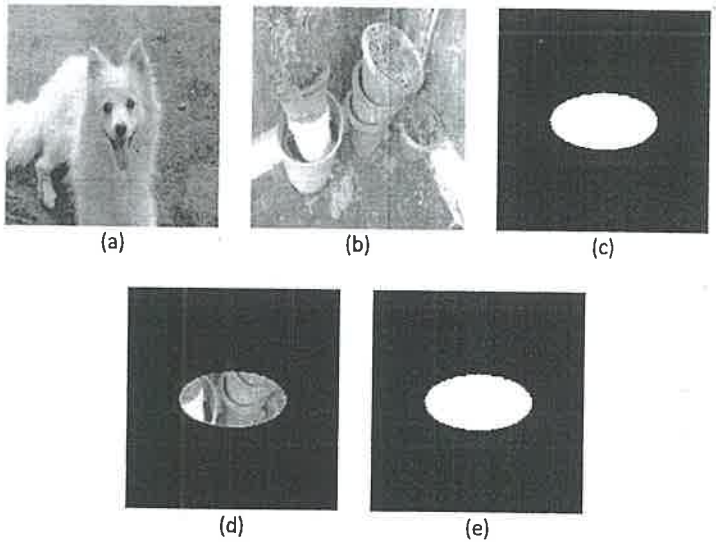


Fig. 3.17 Image division operation (a) Result of the image division operation (image/1.25) (b) Image 1 (c) Image 2 used as a mask (d) Image 3 = image 1 × image 2 (e) Image 4 = image 3/image 1

Example 3.3 Consider the following two images.

$$f_1 = \begin{pmatrix} 1 & 3 & 7 \\ 5 & 15 & 75 \\ 200 & 50 & 150 \end{pmatrix} \quad f_2 = \begin{pmatrix} 50 & 150 & 125 \\ 45 & 55 & 155 \\ 200 & 50 & 75 \end{pmatrix}$$

Perform addition, subtraction, multiplication, division, and image blending operations. Assume both the images are of the 8-bit integer type (uint8 of MATLAB type).

Solution

Addition:

$$g = f_1 + f_2 = \begin{pmatrix} 1+50 & 3+150 & 7+125 \\ 5+45 & 15+55 & 75+155 \\ 200+200 & 50+50 & 150+75 \end{pmatrix} = \begin{pmatrix} 51 & 153 & 132 \\ 50 & 70 & 230 \\ 400 & 100 & 225 \end{pmatrix}$$

If the data type unit8 is assumed, the minimum and maximum allowed values are 0 and 255, respectively. So if the value is larger than 255, it is reset to 255. Similarly, if the value is less than 0, it is reset to 0 (Table 3.1).

So the result of the image addition is

$$g = \begin{pmatrix} 51 & 153 & 132 \\ 50 & 70 & 230 \\ 255 & 100 & 225 \end{pmatrix}$$

Subtraction:

$$g = f_1 - f_2 = \begin{pmatrix} 1-50 & 3-150 & 7-125 \\ 5-45 & 15-55 & 75-155 \\ 200-200 & 50-50 & 150-75 \end{pmatrix} = \begin{pmatrix} -49 & -147 & -118 \\ -40 & -40 & -80 \\ 0 & 0 & 75 \end{pmatrix}$$

Since the data type is unit8, values less than 0 are reset to 0.

$$g = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 75 \end{pmatrix}$$

It can be observed that the modulus of the difference results in a different image.

Multiplication:

$$g = f_1 \times f_2 = \begin{pmatrix} 1\times 50 & 3\times 150 & 7\times 125 \\ 5\times 45 & 15\times 55 & 75\times 155 \\ 200\times 200 & 50\times 50 & 150\times 75 \end{pmatrix} = \begin{pmatrix} 50 & 450 & 875 \\ 225 & 825 & 11625 \\ 40000 & 2500 & 11250 \end{pmatrix}$$

Since the data type is unit8, values greater than 255 are reset to 255.

$$g = \begin{pmatrix} 50 & 255 & 255 \\ 225 & 255 & 255 \\ 255 & 255 & 255 \end{pmatrix}$$

Division:

$$g = f_1 / f_2 = \begin{pmatrix} 1/50 & 3/150 & 7/125 \\ 5/45 & 15/55 & 75/155 \\ 200/200 & 50/50 & 150/75 \end{pmatrix} = \begin{pmatrix} 0.02 & 0.02 & 0.056 \\ 0.11 & 0.272 & 0.484 \\ 1 & 1 & 2 \end{pmatrix}$$

$$g = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 2 \end{pmatrix}$$

This resultant image is due to truncation.

Image or alpha blending: This operation blends two images of the same size to yield a resultant image. This operation is useful for transparency and compositing. The resultant image is a linear combination of two input images. This can be mathematically stated as

$$g(x, y) = \alpha \times f_1(x, y) + (1 - \alpha) \times f_2(x, y)$$

where $f_1(x, y)$ and $f_2(x, y)$ are two input images and $g(x, y)$ is the output image, and the parameter α is called blending ratio, which determines the influence of each image on the resultant image.

Example 3.4 Consider the following 4×4, 8 level images A and B. Find $A + B$, $A - B$, $A \times B$, and A/B.

$$A = \begin{array}{|c|c|c|c|} \hline 1 & 2 & 3 & 4 \\ \hline 5 & 5 & 6 & 6 \\ \hline 6 & 7 & 6 & 6 \\ \hline 6 & 7 & 2 & 3 \\ \hline \end{array} \qquad B = \begin{array}{|c|c|c|c|} \hline 1 & 3 & 5 & 7 \\ \hline 8 & 7 & 0 & 1 \\ \hline 3 & 5 & 6 & 7 \\ \hline 1 & 3 & 5 & 7 \\ \hline \end{array}$$

Solution The given images are 4×4 and 8 grey level (0 – 7) images. As the grey levels are 0 – 7, any value above 7 is reduced to 7.

$$A + B = \begin{array}{|c|c|c|c|} \hline 1+1 & 2+3 & 3+5 & 4+7 \\ \hline 5+8 & 5+7 & 6+0 & 6+1 \\ \hline 6+3 & 7+5 & 6+6 & 6+7 \\ \hline 6+1 & 7+3 & 2+5 & 3+7 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline 2 & 5 & 8 & 11 \\ \hline 13 & 12 & 6 & 7 \\ \hline 9 & 12 & 12 & 13 \\ \hline 7 & 10 & 7 & 10 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline 2 & 5 & 7 & 7 \\ \hline 7 & 7 & 6 & 7 \\ \hline 7 & 7 & 7 & 7 \\ \hline 7 & 7 & 7 & 7 \\ \hline \end{array}$$

$$A - B = \begin{array}{|c|c|c|c|} \hline 1-1 & 2-3 & 3-5 & 4-7 \\ \hline 5-8 & 5-7 & 6-0 & 6-1 \\ \hline 6-3 & 7-5 & 6-6 & 6-7 \\ \hline 6-1 & 7-3 & 2-5 & 3-7 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline 0 & -1 & -2 & -3 \\ \hline -3 & -2 & 6 & 5 \\ \hline 3 & 2 & 0 & -1 \\ \hline 5 & 4 & -3 & -4 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 6 & 5 \\ \hline 3 & 2 & 0 & 0 \\ \hline 5 & 4 & 0 & 0 \\ \hline \end{array}$$

$$A \times B = \begin{array}{|c|c|c|c|} \hline 1\times 1 & 2\times 3 & 3\times 5 & 4\times 7 \\ \hline 5\times 8 & 5\times 7 & 6\times 0 & 6\times 1 \\ \hline 6\times 3 & 7\times 5 & 6\times 6 & 6\times 7 \\ \hline 6\times 1 & 7\times 3 & 2\times 5 & 3\times 7 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline 1 & 6 & 15 & 28 \\ \hline 40 & 35 & 0 & 6 \\ \hline 18 & 35 & 36 & 42 \\ \hline 6 & 21 & 10 & 21 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline 1 & 6 & 7 & 7 \\ \hline 7 & 7 & 0 & 6 \\ \hline 7 & 7 & 7 & 7 \\ \hline 6 & 7 & 7 & 7 \\ \hline \end{array}$$

$$A/B = \begin{array}{|c|c|c|c|} \hline 1/1 & 2/3 & 3/5 & 4/7 \\ \hline 5/8 & 5/7 & 6/0 & 6/1 \\ \hline 6/3 & 7/5 & 6/6 & 6/7 \\ \hline 6/1 & 7/3 & 2/5 & 3/7 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 6 \\ \hline 2 & 1 & 1 & 0 \\ \hline 6 & 2 & 0 & 0 \\ \hline \end{array}$$

3.2.1.5 Applications of arithmetic operations

Arithmetic operations can be combined and put to effective use. For example, the image averaging process can be used to remove noise. Noise is a random fluctuation of pixel values, which affects the quality of the image. A noisy image can be considered as an image plus noise:

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

where $f(x, y)$ is the input image and $g(x, y)$ is the output image. Several instances of noisy images can be taken and averaged as

$$\bar{g}(x, y) = \frac{1}{M} \sum_{i=1}^{M} g_i(x, y)$$

where M is the number of noisy images. As M increases, the averaging process reduces the intensity of the noise and it becomes so low that it can automatically be removed. As M becomes large, the expectation $E\{\bar{g}(x, y)\} = f(x, y)$.

3.2.2 Logical Operations

Bitwise operations can be applied to image pixels. The resultant pixel is defined by the rules of the particular operation. Some of the logical operations that are widely used in image processing are as follows:

1. AND/NAND
2. OR/NOR
3. EXOR/EXNOR
4. Invert/Logical NOT

3.2.2.1 AND/NAND

The truth table of the AND and NAND operators is given in Table 3.2.

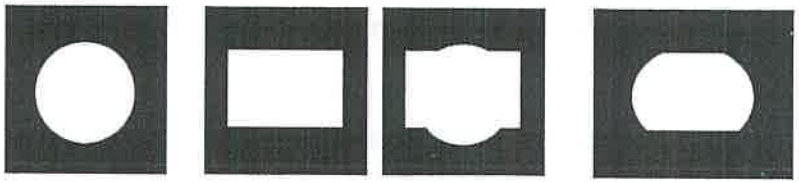
Table 3.2 Truth table of the AND and NAND operators

A	B	C (AND)	C (NAND)
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

The operators AND and NAND take two images as input and produce one output image. The output image pixels are the output of the logical AND/NAND of the individual pixels. Some of the practical applications of the AND and NAND operators are as follows:

1. Computation of the intersection of images
2. Design of filter masks
3. Slicing of grey scale images; for example, the pixel value of the grey scale image may be 1100 0000. The first bits of the pixels of an image constitute one slice. To extract the first slice, a mask of value 1000 0000 can be designed. The AND operation of the image pixel and the mask can extract the first bit and first slice of the image.

Figures 3.18(a)–3.18(d) shows the effects of the AND and OR logical operators. It illustrates that the AND operator shows overlapping regions of the two input images and the OR operator shows all the input images with their overlapping.



(a) (b) (c) (d)

Fig. 3.18 Results of the AND and OR logical operators (a) Image 1 (b) Image 2 (c) Result of image 1 OR image 2 (d) Result of image 1 AND image 2

3.2.2.2 OR/NOR

The truth table of the OR and NOR operators is given in Table 3.3.

Table 3.3 Truth table of the OR and NOR operators

A	B	C (OR)	C (NOR)
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

The practical applications of the OR and NOR operators are as follows:

1. OR is used as the union operator of two images.
2. OR can be used as a merging operator.

3.2.2.3 XOR/XNOR

The truth table of the XOR and XNOR operators is given in Table 3.4.

Table 3.4 Truth table of the XOR and XNOR operators

A	B	C (XOR)	C (XNOR)
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

The practical applications of the XOR and XNOR operators are as follows:

1. Change detection
2. Use as a subcomponent of a complex imaging operation. XOR for identical inputs is zero. Hence it can be observed that the common region of image 1 and image 2 in Figs 3.18(a) and 3.18(b), respectively, is zero and hence dark. This is illustrated in Fig. 3.19.

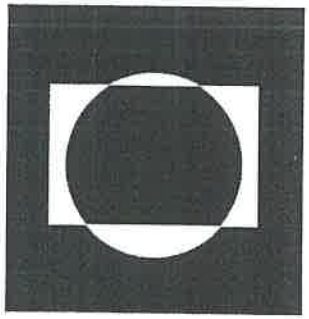


Fig. 3.19 Result of the XOR operation

3.2.2.4 Invert/Logical NOT

The truth table of the NOT operator is given in Table 3.5.

For grey scale values, the inversion operation is described as

$$g(x, y) = 255 - f(x, y)$$

The practical applications of the inversion operator are as follows:

1. Obtaining the negative of an image. Figure 3.20 shows the negative of the original image shown in Fig. 3.18(a).
2. Making features clear to the observer
3. Morphological processing

Table 3.5 Truth table of the NOT operator

A	C (NOT)
0	1
1	0

Similarly, two images can be compared using operators such as

= Equal to

> Greater than

≥ Greater than or equal to

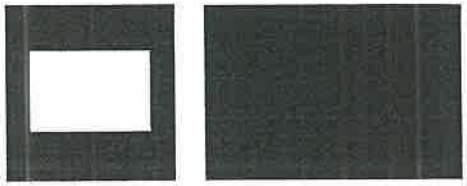
< Less than

≤ Less than or equal to

≠ Not equal to

The resultant image pixel represents the truth or falsehood of the comparisons. Similarly, shifting operations are also very useful. Shifting of I bits of the image pixel to the right results in division by 2^I. Similarly, shifting of I bits of the image pixel to the left results in multiplication by 2^I.

Shifting operators are helpful in dividing and multiplying an image by a power of two. In addition, this operation is computationally less expensive.



(a) (b)

Fig. 3.20 Effect of the NOT operator (a) Original Image (b) NOT of original Image

Example 3.5 Consider the following two images.

$$f_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \quad f_2 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Perform the logical AND, OR, NOT, and difference operations.

Solution

AND

$$f_1 \text{ AND } f_2 = \begin{pmatrix} 1 \wedge 1 & 0 \wedge 1 & 0 \wedge 1 \\ 1 \wedge 1 & 1 \wedge 1 & 1 \wedge 1 \\ 0 \wedge 1 & 0 \wedge 1 & 1 \wedge 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

OR

$$f_1 \text{ OR } f_2 = \begin{pmatrix} 1 \vee 1 & 0 \vee 1 & 0 \vee 1 \\ 1 \vee 1 & 1 \vee 1 & 1 \vee 1 \\ 0 \vee 1 & 0 \vee 1 & 1 \vee 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

NOT

$$\text{NOT}(f_1) = \begin{pmatrix} \neg 1 & \neg 0 & \neg 0 \\ \neg 1 & \neg 1 & \neg 1 \\ \neg 0 & \neg 0 & \neg 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

Difference

$$f_1 \text{ AND } (\neg f_2) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

and

$$f_2 \text{ AND } (\neg f_1) = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

3.2.3 Geometrical Operations

Let us now discuss the geometrical operations used in image processing.

3.2.3.1 *Translation*

Translation is the movement of an image to a new position. Let us assume that the point at the coordinate position $X = (x, y)$ of the matrix $\boldsymbol{F}$ is moved to the new position X' whose coordinate position is (x', y'). Mathematically, this can be stated as a translation of a point X to the new position X'. The translation is represented as

$$x' = x + \delta x$$

$$y' = y + \delta y$$

The translation of the floor image by (25, 25) is shown in Fig. 3.21. In vector notation, this is represented as $\boldsymbol{F'} = \boldsymbol{F} + T$, where δx and δy are translations parallel to the x and y axes. Here, $\boldsymbol{F}$ and $\boldsymbol{F'}$ are the original and the translated images, respectively. However, other transformations such as scaling and rotation are multiplicative in nature. The transformation process for rotation is given as $\boldsymbol{F'} = \boldsymbol{RF}$, where $\boldsymbol{R}$ is the transform matrix for performing rotation, and the transformation process for scaling is given as $\boldsymbol{F'} = \boldsymbol{SF}$. Here, $\boldsymbol{S}$ is the scaling transformation matrix.

Fig. 3.21 Result of translation by 50 units

To create uniformity and consistency, it is necessary to use a homogeneous coordinate system where all transformations are treated as multiplications. A point (x, y) in 2D space is expressed as (wx, wy, w) for $w \neq 0$. The properties of homogeneous coordinates are as follows:

1. In homogeneous coordinates, at least one point should be non-zero. Thus (0, 0, 0) does not exist in the homogeneous coordinate system.
2. If one point is multiplicative of the other point, they are same. Thus, the points (1, 3, 5) and (3, 9, 15) are same as the second point is $3 \times (1, 3, 5)$.
3. The point (x, y, w) in the homogenous coordinate system corresponds to the point $\left(\frac{x}{w}, \frac{y}{w}\right)$ in 2D space.

In the homogeneous coordinate system, the translation process of point (x, y) of image F′ to the new point (x', y') of the image F′ is described as

$$x' = x + \delta x$$

and

$$y' = y + \delta y$$

In matrix form, this can be stated as

$$[x', y', 1] = \begin{pmatrix} 1 & 0 & \delta x \\ 0 & 1 & \delta y \\ 0 & 0 & 1 \end{pmatrix} [x, y, 1]^T$$

Sometimes, the image may not be present at the origin. In that case, a suitable negative translational value can be used to bring the image to align with the origin.

3.2.3.2 Scaling

Depending on the requirement, the object can be scaled. *Scaling* means enlarging and shrinking. In the homogeneous coordinate system, the scaling of the point (x, y) of the image F to the new point (x', y') of the image F′ is described as

$$x' = x \times S_x$$
$$y' = y \times S_y$$

$$[x', y'] = \begin{pmatrix} S_x & 0 \\ 0 & S_y \end{pmatrix} [x, y]$$

S_x and S_y are called scaling factors along the x and y axes, respectively. If the scale factor is 1, the object would appear larger. If the scaling factors are fractions, the object would shrink. Similarly, if S_x and S_y are equal, scaling is uniform. This is known as *isotropic scaling*. Otherwise, it is called *differential scaling*. In the homogeneous coordinates system, it is represented as

$$[x', y', 1] = \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{pmatrix} [x, y, 1]^T$$

The matrix $S = \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$ is called scaling matrix.

3.2.3.3 Mirror or reflection operation

This function creates the reflection of the object in a plane mirror. The function returns an image in which the pixels are reversed. This operation is useful in creating an image in the desired order and for making comparisons. The reflected object is of the same size as the original object, but the object is in the opposite quadrant. Reflection is also described as rotation by 180°. The reflection along the x-axis is given by

$$F' = [-x, y] = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \times [x, y]^T$$

Similarly, the reflection along the y-axis is given by

$$F' = [x, -y] = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \times [x, y]^T$$

Similarly, the reflection about the line $y = x$ is given as

$$F' = [x, -y] = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \times [x, y]^T$$

(a) (b)

Fig. 3.22 Mirror operation (a) Original image (b) Reflection along the y-axis

The reflection about $y = -x$ is given as

$$F' = [x, -y] = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} \times [x, y]^T$$

The reflection operation is illustrated in Figs 3.22(a) and 3.22(b). In the homogeneous coordinate system, the matrices for reflection can be given as

$$R_{y\text{-axis}} = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}; R_{x\text{-axis}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}; R_{\text{origin}} = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Similarly, the reflection along a line can be given as

$$R_{y=x} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}; R_{y=-x} = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

3.2.3.4 Shearing

Shearing is a transformation that produces a distortion of shape. This can be applied either in the x-direction or the y-direction. In this transformation, the parallel and opposite layers of the object are simply sided with respect to each other.

Shearing can be done using the following calculation and can be represented in the matrix form as

$$x' = x + ay$$

$$y' = y$$

$$X_{\text{shear}} = \begin{pmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ (where } a = sh_x\text{)}$$

Similarly, Y_{shear} can be given as

$$x' = x$$

$$y' = y + bx$$

$$Y_{\text{shear}} = \begin{pmatrix} 1 & 0 & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ (where } b = sh_y\text{)}$$

where sh_x and sh_y are shear factors in the x and y directions, respectively.

3.2.3.5 Rotation

An image can be rotated by various degrees such as 90°, 180°, or 270°. In the matrix form, it is given as

$$[x', y'] = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} [x, y]^T$$

This can be represented as $F' = RA$. The parameter θ is the angle of rotation with respect to the x-axis. It is assumed that the object rotation is about the origin. The value of θ can be positive or negative. A positive angle represents counter clockwise rotation and a negative angle represents clockwise rotation. The rotation of an image is shown in Figs 3.23(a) and 3.23(b). In the homogeneous coordinate system, rotation can be represented as

$$[x', y', 1] = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} [x, y, 1]^T$$

If θ is substituted with $-\theta$, this matrix rotates the image in the clockwise direction.

(a) (b)

Fig. 3.23 Rotation (a) Original image (b) Result of rotation by 45°

3.2.3.6 Affine transform

The transformation that maps the pixel at the coordinates (x, y) to a new coordinate position is given as a pair of transformation equations. In this transform, straight lines are preserved and parallel lines remain unchanged. It is described mathematically as

$$x' = T_x(x, y)$$

$$y' = T_y(x, y)$$

T_x and T_y are expressed as polynomials. The linear equation gives an affine transform.

$$x' = a_0 x + a_1 y + a_2$$

$$y' = b_0 x + b_1 y + b_2$$

This is expressed in matrix form as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

The affine transform is a compact way of representing all transformations. The given equation represents all transformations. Translation is the situation where $a_0 = 1$, $a_1 = 0$, and $a_2 = \delta_x$; scaling transformation is a situation where $a_0 = s_x$ and $b_1 = s_y$ and $a_1 = 0$, $a_2 = 0$, $b_0 = 0$, and $b_2 = 0$; rotation is a situation where $a_0 = \cos\theta$, $a_1 = -\sin\theta$, $b_0 = \sin\theta$, $b_1 = \cos\theta$, $a_2 = 0$, and $b_2 = 0$; and horizontal shear is performed when $a_0 = 1$, $b_0 = 1$, $a_1 = Sh_x$, $a_2 = 0$, $b_1 = Sh_y$ and $b_2 = 0$.

3.2.3.7 *Inverse transformation*

The purpose of inverse transformation is to restore the transformed object to its original form and position. The inverse or backward transformation matrices are given as follows:

$$\text{Inverse transform for translation} = \begin{pmatrix} 1 & 0 & -\delta x \\ 0 & 1 & -\delta y \\ 0 & 0 & 1 \end{pmatrix}$$

$$\text{Inverse transform for scaling} = \begin{pmatrix} \frac{1}{S_x} & 0 & 0 \\ 0 & \frac{1}{S_y} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Inverse transform for rotation can be obtained by changing the sign of the transform term. For example, the following matrix performs inverse transform.

$$\begin{pmatrix} \cos\theta & +\sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Example 3.6 Consider an image point [2, 2]. Perform the following operations and show the results of these transforms.

(a) Translate the image right by 3 units.

(b) Perform a scaling operation in both x-axis and y-axis by 3 units.

(c) Rotate the image in x-axis by 45°.

(d) Perform horizontal skewing by 45°.

(e) Perform mirroring about x-axis.

(f) Perform shear in y-direction by 30 units.

Solution

(a) Translation of the image right by 3 units means that

$$\delta_x = 3 \text{ and } \delta_y = 0$$

So the translation matrix is given as

$$\boldsymbol{T} = \begin{pmatrix} 1 & 0 & \delta_x \\ 0 & 1 & \delta_y \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Therefore,

$$x' = \boldsymbol{T} \times x$$

$$= \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times [2, 2, 1] = [5 \quad 2 \quad 1]^T$$

(b) Scaling by 3 units in both the directions means that

$$\boldsymbol{S} = \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\boldsymbol{S} = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times [2, 2, 1]^T = [6 \quad 6 \quad 1]^T$$

(c) Rotating the image in x-axis by 45°

$$\boldsymbol{R} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos 45 & -\sin 45 & 0 \\ \sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\boldsymbol{R} = \begin{pmatrix} 0.707 & -0.707 & 0 \\ 0.707 & 0.707 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times [2, 2, 1]^T = [0 \quad 2.828 \quad 1]^T$$

The fraction 2.828 will be rounded to 3. This is determined by the interpolation technique used.

(d) Performing horizontal skewing by 45°

$$Skew = \begin{pmatrix} 1 & 0 & 0 \\ \tan\varphi & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ \tan 45 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$Skew = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times [2, 2, 1]^T = [2, 4, 1]^T$$

(e) Performing mirroring about x-axis

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

$$Mirroring = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times [2, 2, 1] = [2 \quad -2 \quad 1]^T$$

(f) Performing shear in y-direction by 30 units

$$Shear_y = \begin{pmatrix} 1 & Sh_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 30 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$Mirroring = \begin{pmatrix} 1 & 30 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times [2, 2, 1]^T = [62 \quad 2 \quad 1]^T$$

3.2.3.8 3D Transforms

Some medical images such as computerized tomography (CT) and magnetic resonance imaging (MRI) images are three-dimensional images. To apply translation, rotation, and scaling on 3D images, 3D transformations are required. 3D transformations are logical extensions of 2D transformations. These are summarized and described as follows:

$$\text{Translation} = \begin{pmatrix} 0 & 0 & 0 & \delta x \\ 0 & 1 & 0 & \delta y \\ 0 & 0 & 1 & \delta z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\text{Scaling} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation along z-axis

$$R_{z,\theta} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation along x-axis

$$R_{x,\theta} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation along y-axis

$$R_{y,\theta} = \begin{pmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Compactly, these can be represented as

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = C \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Similarly, the reflection transform matrices can be given as

$$\text{Reflection}_{xy\text{-plane}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\text{Reflection}_{xz\text{-plane}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\text{Reflection}_{yz\text{-plane}} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The shear matrices for the shear quantities a and b can be given as

$$\text{Shear}_z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ a & b & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\text{Shear}_x = \begin{pmatrix} 1 & a & b & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\text{Shear}_y = \begin{pmatrix} 1 & 0 & 0 & 0 \\ a & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

3.2.4 Image Interpolation Techniques

What is the need for interpolation techniques? Transforms can be of two types. Affine transforms often produce pixels of the resultant image that cannot be fit as some of the pixel values are non-integers and often go beyond the acceptable range. This results in gaps (or holes) and issues related to number of pixels and range. So interpolation techniques are required to solve these issues.

3.2.4.1 Forward mapping

Forward mapping is the process of applying transformations iteratively to every pixel in the image, yielding a new coordinate position and copying the values of the pixel to a new position.

3.2.4.2 Backward mapping

Similarly, backward mapping is the process of checking the pixels of the output image to determine the position of the pixels in the input image. This is used to guarantee that all the pixels of the input image are processed.

During the process of both forward and backward mapping, it may happen that pixels cannot be fitted into the new coordinates. For example, consider the process of rotation of a point, (10, 5), by 45°. This yields

$$\begin{aligned} x' &= x\cos\theta - y\sin\theta \\ &= 10\cos(45°) - 5\sin(45°) \\ &= 10(0.707) - 5(0.707) \\ &= 3.535 \\ y' &= x\sin\theta + y\cos\theta \\ &= 10\sin(45°) + 5\cos(45°) \\ &= 10(0.707) + 5(0.707) \\ &= 10.605 \end{aligned}$$

Since these new coordinate positions are not integers, the rotation process cannot be carried out. Thus, the process may leave a gap in the new coordinate position, which creates poor quality output. Therefore, whenever a geometric transformation is performed, a resampling process should be carried out so that the desirable quality is achieved in the resultant image. The resampling process creates new pixels so that the quality of the output

is maintained. In addition, the rounding off of the new coordinate position (3.535, 10.605) should be carried out as (4, 11). This process of fitting the output to the new coordinates is called interpolation.

Interpolation is the method of calculating the expected values for a function with known pixels. Some of the popular interpolation techniques are

1. Nearest neighbour technique
2. Bilinear technique
3. Bicubic technique

The most elementary form of interpolation is nearest neighbour interpolation or zero-order interpolation. This technique determines the closest pixel and assigns it to every pixel in the new image matrix, that is, the brightness of the pixels is equal to the closest neighbour. Sometimes, this may result in pixel blocking and can degrade the resulting image, which may appear spatially distorted. These distortions are called aliasing.

A more accurate interpolation scheme is bilinear interpolation. This is called first-order interpolation. Here, four neighbours of the transformed original pixels that surround the new pixel are obtained and are used to calculate the new pixel value. Linear interpolation is used in both the directions. Weights are assigned based on the proximity. Then the process takes the weighted average of the brightness of the four pixels that surround the pixels of interest.

$$\begin{aligned} g(x, y) = (1-a)(1-b)f(x', y') + (1-a)\,b\,f(x', y'+1) \\ + a\,(1-b)f(x'+1, y') + a\,b\,f(x'+1, y'+1) \end{aligned}$$

Here, g(x, y) is the output image and f(x, y) is the image that undergoes the interpolation operation. If the desired pixel is very close to one of the four nearest neighbour pixels, its weight will be much higher. This technique leads to blurring of the edges. However, it reduces aliasing artefacts.

High-order interpolation scheme takes more pixels into account. Second-order interpolation is known as cubic interpolation; it uses a neighbourhood of 16 pixels. Then it fits two polynomials to the 16 pixels of the transformed original matrix and the new image pixel. This technique is very effective and produces images that are very close to the original. In extreme cases, more than 64 neighbouring pixels can be used. However, as the number of pixels increases, the computational complexity also increases. The results of interpolation are shown in Fig. 3.24.

(a) (b) (c)

Fig. 3.24 Result of interpolation (a) Nearest neighbour (b) Bilinear (c) Bicubic

3.2.5 Set Operations

An image can be visualized as a set. For example, the following binary image (Fig. 3.25) can be visualized as a set A, where $A = \{(0, 0), (0, 2), (2, 2)\}$. The coordinates' values represent the value of 1. Set operators can then be applied to the set to get the resultant, which is useful for image analysis.

F:

	0	1	2
0	1	0	1
1	0	0	0
2	0	0	1

Fig. 3.25 Sample binary image

The complement of set A can be defined as the set of pixels that does not belong to the set A.

$$A^c = \{c/c \notin A\}$$

The reflection of the set is defined as

$$A = \{c = -a, a \in A\}$$

The union of two sets, A and B, can be represented as

$$A \cup B = \{c/(c \in A) \vee (c \in B)\}$$

where the pixel c belongs to A, B, or both.

The intersection of two sets is given as $A \cap B = \{c/(c \in A) \wedge (c \in B)\}$. The pixel c belongs to A, B, or both.

The difference can be expressed as

$$A - B = \{c/(c \in A) \wedge (c \notin B)\}$$

which is equivalent to $A \cap B^c$.

Morphology is a collection of operations based on set theory, to accomplish various tasks such as extracting boundaries, filling small holes present in the image, and removing noise present in the image.

Mathematical morphology is a very powerful tool for analysing the shapes of the objects that are present in the images. The theory of mathematical morphology is based on set theory. One can visualize a binary object as a set. Set theory can then be applied to the sample set. Morphological operators often take a binary image and a mask known as structuring element as input. The set operators such as intersection, union, inclusion, and complement can then be applied to the images. Dilation is one of the two basic operators. It can be applied to binary as well as grey scale images. The basic effect of this operator on a binary image is that it gradually increases the boundaries of the region, while the small holes that are present in the images become smaller.

Let us assume that A and B are a set of pixel coordinates. The dilation of A by B can be denoted as

$$A \oplus B = \{(x, y) + (u, v): (x, y) \in A, (u, v) \in B\}$$

where x and y correspond to the set A, and u and v correspond to the set B. The coordinates are added and the union is carried out to create the resultant set. These kinds of operations are based on Minkowski algebra.

Example 3.7 Consider the following binary image. Show the results of the dilation and erosion operations.

F:

	0	1	2
0	0	0	1
1	0	0	1
2	0	1	1

Let the structured element S be [1 1] with coordinates $\{(0, 0), (0, 1)\}$. Show the results of dilation and erosion.

Solution The image F can be written as

$$F = \{(0, 2), (1, 2), (2, 1), (2, 2)\}$$
$$S = \{(0, 0), (0, 1)\}$$

The dilation operation is done as follows:

First add the coordinates (0, 0) of S to all the coordinate points of the image set F, followed by the second point of the set S.

$$F \text{ Dilation } S = \{(0, 2), (1, 2), (2, 1), (2, 2), (0, 3), (1, 3), (2, 2), (2, 3)\}$$

Remove the repetitions; the union of these sets results in dilation. This results in

$$S = \{(0, 2), (0, 3), (1, 2), (1, 3), (2, 1), (2, 2), (2, 3)\}$$

The erosion is the intersection of these sets.

First subtract the coordinates (0, 0) of S from all the coordinate points of the image set F, followed by the second point of the set S.

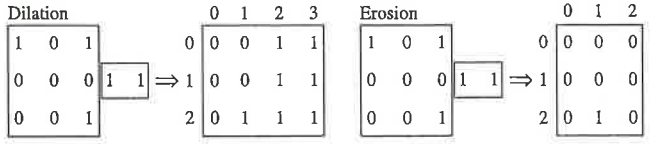
$$F \text{ Erosion } S = \{(0, 2), (1, 2), (2, 1), (2, 2), (0, 1), (1, 1), (2, 0), (2, 1)\}$$

The erosion is the intersection operation. Find the common element. This results in

$$S = (2, 1)$$

If the coordinates are (x, y) and (s, t), the result would be $(x + s, y + t)$ for dilation and $(x - s, y - t)$ for erosion.

The result of this numerical calculation is shown in Fig. 3.26(a).



(a)

(b) (c) (d) (e)

Fig. 3.26 Dilation operation (a) Effects of dilation and erosion for a numerical example (b) Original large image (c) Dilation operation with structural element (of order 3 × 3) (d) Dilation operation with structural element (of order 9 × 9) (e) Dilation operation with structural element (of order 13 × 13)

A dilation process for the bigger images requires a lot of computational effort. For example, the image shown in Fig. 3.26(b) illustrates one application of the dilation operator. The dilation operation with three sets of structural elements is used in Figs 3.26(c)–3.26(e). It can be observed that the brightness of the image increases.
The morphological operators are covered in detail in Chapter 9.

3.2.6 Statistical Operations

Statistics play an important role in image processing. An image can be assumed to be a set of discrete points. Statistical operations can be applied to the image to get the desired results such as manipulation of brightness and contrast. Some of the very useful statistical operations include mean, median, mode, and mid-range. These measures are useful in image processing. The measures of data dispersion also include quartiles, inter-quartile range, and variance.

Some of the frequently used statistical measures are the following:

Mean Mean is the average of all the values in the sample (population) and is denoted as $\bar{X}$.

$$\bar{X} = \frac{X_1 + X_2 + \cdots + X_n}{n} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

The overall brightness of the grey scale image is measured using the mean. This is calculated by summing all the values of the pixels of an image and dividing it by the number of pixels in the image.

$$\mu = \frac{1}{n}\sum_{i=0}^{n-1} I_i$$

Sometimes the data is associated with a weight. This is called weighted mean. The problem of mean is its extreme sensitivity to noise. Even small changes in the input affect the mean drastically.

Median Median is the value where the given X_i is divided into two equal halves, with half of the values being lower than the median and the other half higher. The procedure for obtaining the median is to sort the values of the given X_i in ascending order. If the given sequence has an odd number of values, the middle value is the median. Otherwise, the median is the arithmetic mean of the two middle values.

Mode Mode is the value that occurs most frequently in the dataset. The procedure for finding the mode is to calculate the frequencies for all of the values in the data. The mode is the value (or values) with the highest frequency. Normally, based on the mode, the dataset is classified as unimodal, bimodal, and trimodal. Any dataset that has two modes is called bimodal.

Percentile Percentiles are data that are less than the coordinate by some percentage of the total value. For example, the median is the 50th percentile and can be denoted as $Q_{0.50}$. The 25th percentile is called the first quartile and the 75th percentile is called third quartile. Another measure that is useful to measure dispersion is the inter-quartile range. The inter-quartile is defined as $Q_{0.75}$ $Q_{0.25}$. Semi quartile range is = 0.5 × IQR.

Unimodal curves are slightly skewed and the empirical relation is

$$\text{Mean} - \text{Mode} = 3 \times (\text{Mean} - \text{Median})$$

The interpretation of the formula is that the mode for the unimodal frequency curve is moderately skewed. The mid-range is also used to assess the central tendency of the dataset. In a normal distribution, the mean, median, and mode are the same. In symmetrical distributions, it is possible for the mean and median to be the same even though there may be several modes. By contrast, in asymmetrical distributions, the mean and median are not the same. These distributions are said to be skewed data where more than half the cases are either above or below the mean.

Standard deviation and variance By far, the most commonly used measures of dispersion are variance and standard deviation. The mean does not convey much more than a middle point. For example, the following datasets, {10, 20, 30} and {10, 50, 0}, both have a mean of 20. The difference between these two sets is the spread of data. Standard deviation is the average distance from the mean of the dataset to each point. The formula for standard deviation is given by

$$\sigma = \sqrt{\sum_{i=1}^{N} (Xi - \mu)^2 / N - 1}$$

Sometimes, we divide the value by $N - 1$ instead of N. The reason is that in a larger, real-world scenario, division by $N - 1$ gives an answer that is closer to the actual value. In image processing, it is a measure of how much a pixel varies from the mean value of the image. The mean value and the standard deviation characterize the perceived brightness and contrast of the image. Variance is another measure of the spread of the data. It is the square of standard deviation. While standard deviation is a more common measure, variance also indicates the spread of the data effectively.

Entropy This is the measure of the amount of orderliness that is present in the image. The entropy can be calculated by assuming that the pixels are totally uncorrelated. An organized system has low entropy and a complex system has a very high entropy. Entropy also indicates the average global information content. Its unit is bits per pixel. It can be computed using the formula

$$\text{Entropy } H = -\sum_{i=1}^{n} p_i \log_2 p_i$$

where p_i is the prior probability of the occurrence of the message. Let us consider a binary image, where the pixel assumes only two possible states, 0 or 1 and the occurrence of each

state is equally likely. Hence, the probability is 1/2. Therefore, the entropy is $H = -[1/2 \log (1/2) + 1/2 \log (1/2)] = 1$ bit. Therefore, 1 bit is sufficient to store the intensity of the pixel. Therefore, binary images are less complex.

Example 3.8 Calculate the entropy for the symbols given in Table 3.6.

Table 3.6 Symbols and their probability

Symbols	1	2	3	4	5
Probability	0.4	0.3	0.1	0.1	0.1

$$\text{Entropy } H = -\sum_{i=1}^{n} p_i \log_2 p_i$$

$$H = -(0.4\log_2 0.4 + 0.3\log_2 0.3 + 0.1\log_2 0.1 + 0.1\log_2 0.1 + 0.1\log_2 0.1)$$

$$H = -(-0.5288 - 0.5211 - 0.3322 - 0.3322 - 0.3322)$$
$$= 2.0465$$

Example 3.9 Calculate the entropy of the given image

1	2	3	5
3	4	4	6
9	9	8	7
1	3	5	6

Solution The formula for computing the entropy is

$$\text{Entropy} = -\sum_{i=1}^{n} p_i \log p_i .$$

The probability of all the pixels are computed and shown in the following table.

Symbol in image	Probability of occurrence (p_i)
1	2/16
2	1/16
3	3/16
4	2/16
5	2/16
6	2/16
7	1/16
8	1/16
9	2/16

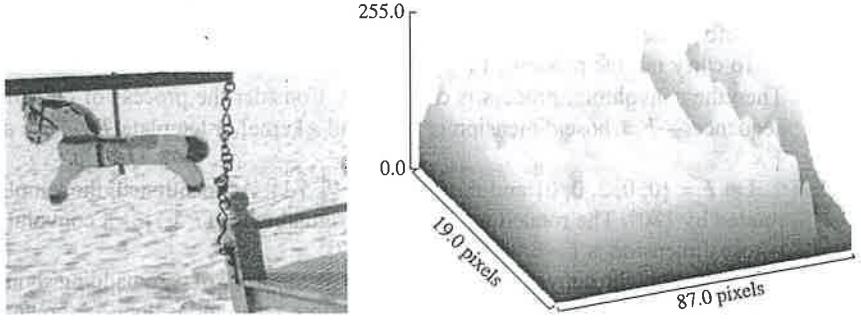
Therefore, the entropy is computed as follows:

$$\text{Entropy} = -\left\{\frac{2}{16}\log\left(\frac{2}{16}\right)+\frac{1}{16}\log\left(\frac{1}{16}\right)+\frac{3}{16}\log\left(\frac{3}{16}\right)+\frac{2}{16}\log\left(\frac{2}{16}\right)+\frac{2}{16}\log\left(\frac{2}{16}\right)+\frac{2}{16}\log\left(\frac{2}{16}\right)+\frac{1}{16}\log\left(\frac{1}{16}\right)+\frac{1}{16}\log\left(\frac{1}{16}\right)+\frac{2}{16}\log\left(\frac{2}{16}\right)\right\}$$

$$= -\{-0.375 - 0.25 - 0.45 - 0.375 - 0.375 - 0.375 - 0.25 - 0.25 - 0.375\}$$
$$= -\{-3.075\}$$

Therefore, the entropy of the image is 3.075.

Thus, entropy indicates the information richness of the image. This can be seen visually using a surface plot where pixel values are plotted as a function of pixel position. One such plot is shown in Figs 3.27(a) and 3.27(b). Entropy can also be used to characterize the texture of the images.



(a) (b)

Fig. 3.27 Surface plot (a) Original image (b) Surface plot

Applications of statistical operations The brightness and contrast can be defined as the mean grey level and the mean absolute deviation (MAD), respectively. It is given as

$$\text{MAD} = 1/MN \times \sum_{i=1}^{M} \sum_{i=1}^{N} |(x, y) = mean|$$

Then the equation

$$g(x, y) = \left(f(x, y)/\frac{\sigma}{\sigma_n} \right) + \mu_n$$

can be used with σ_n and μ_n, which are the standard deviation and mean of the desired new image. M and N are the dimensions of the image. Generally, this operation increases the brightness and contrast.

3.2.7 Convolution and Correlation Operations

The imaging system can be modelled as a 2D linear system. Let $f(x, y)$ and $g(x, y)$ represent the input and output images, respectively. Then, they can be written as $g(x, y) = t * (f(x, y)$. Convolution is a group process, that is, unlike point operations, group processes operate on a group of input pixels to yield the result. Spatial convolution is a method of taking a group of pixels in the input image and computing the resultant output image. This is also known as a finite impulse response (FIR) filter. Spatial convolution moves across pixel by pixel and produces the output image. Each pixel of the resultant image is dependent on a group of pixels (called kernel).

WORD SEARCH PUZZLE

Some of the important terms in this chapter are present in the following word jumble. Identify the words. Diagonal words are possible.

J	A	S	A	C	M	C	T	N	H	B	P	A	K	N	S	P	B	U	P	V	W	H	I	Y
J	I	M	K	Z	O	E	V	O	I	F	Q	V	E	E	Y	W	L	D	Y	P	A	R	R	P
X	S	K	R	V	Q	K	F	Z	D	E	T	A	I	L	F	J	C	R	E	T	Y	A	L	R
N	U	N	H	E	X	V	R	Q	W	W	K	F	B	T	U	U	S	H	S	I	T	O	I	E
M	U	L	T	I	R	E	S	O	L	U	T	I	O	N	H	E	T	W	I	N	C	N	W	C
N	C	H	Z	E	G	A	G	P	R	F	I	I	I	Z	Y	J	U	L	E	H	J	H	C	J
N	U	W	A	H	R	N	K	X	A	C	B	U	O	F	O	L	M	M	Z	M	Q	O	G	C
I	L	F	A	K	S	Z	L	Z	E	X	A	L	E	K	G	P	E	T	C	Z	B	V	E	F
N	C	H	X	E	F	A	V	L	A	P	S	E	Y	P	F	L	P	S	B	Q	R	D	H	H
Z	Z	N	Y	M	C	R	A	P	S	S	N	H	X	V	P	Z	W	O	X	F	Z	C	R	Z
U	U	O	F	O	C	C	M	L	L	E	O	F	T	M	W	Q	R	G	Z	W	S	L	G	B
V	Y	U	L	B	S	U	P	L	A	P	P	R	O	X	I	M	A	T	I	O	N	O	D	S
B	E	I	D	E	N	Z	B	I	R	T	G	C	N	L	Z	J	V	N	E	M	Y	J	W	K
L	S	N	R	P	S	M	E	X	P	A	N	S	I	O	N	H	I	X	U	S	U	K	D	L
E	G	J	J	W	T	H	G	C	R	W	E	A	T	A	G	U	M	W	A	V	E	X	Q	H
R	R	A	L	S	B	Y	Y	E	S	O	C	S	I	K	U	E	S	C	A	L	E	P	A	Q
Z	V	J	T	U	J	Q	Y	Y	E	S	C	A	L	I	N	G	V	F	E	I	P	T	G	F

Hints

1. Examining images at multiple resolutions is called _____ analysis.
2. Wavelet means a small _____.
3. Wavelet transforms are useful for analysing _____ portions of the image.
4. _____ and _____ coefficients are the outputs of a wavelet transform.
5. Vector spaces and its _____ vector spaces are used in wavelet expansion.
6. Wavelet _____ is the basis of wavelet transform.
7. Frequency is often called a _____.
8. Dilation is a _____ of signals.
9. Wavelet is subjected to _____ for picking out variability in different time scales.

CHAPTER 9

Image Segmentation

Not everything that can be counted counts, and not everything that counts can be counted.
–Albert Einstein

LEARNING OBJECTIVES

Segmentation is the process of partitioning a digital image into multiple regions and extracting meaningful regions known as regions of interest (ROI) for further image analysis. Segmentation is an important phase in the image analysis process. After studying this chapter, the reader will become familiar with the following:

- Types of segmentation algorithms
- Edge detection algorithms
- Thresholding techniques
- Region-oriented segmentation algorithms
- Active contour models
- Evaluation of segmentation algorithms

9.1 INTRODUCTION

Image segmentation has emerged as an important phase in image-based applications. Segmentation is the process of partitioning a digital image into multiple regions and extracting a meaningful region known as the region of interest (ROI). Regions of interest vary with applications. For example, if the goal of a doctor is to analyse the tumour in a computer tomography (CT) image, then the tumour in the image is the ROI. Similarly, if the image application aims to recognize the iris in an eye image, then the iris in the eye image is the required ROI. Segmentation of ROI in real-world images is the first major hurdle for effective implementation of image processing applications as the segmentation process is often difficult. Hence, the success or failure of the extraction of ROI ultimately influences the success of image processing applications. No single universal segmentation algorithm exists for segmenting the ROI in all images. Therefore, the user has to try many segmentation algorithms and pick an algorithm that performs the best for the given requirement.

Image segmentation algorithms are based on either discontinuity principle or similarity principle. The idea behind discontinuity principle is to extract regions that differ in properties such as intensity, colour, texture, or any other image statistics. Mostly, abrupt changes in intensity among the regions result in extraction of edges. The idea behind similarity principle is to group pixels based on a common property, to extract a coherent region.

9.1.1 Formal Definition of Image Segmentation

An image can be partitioned into many regions $R_1, R_2, R_3, \ldots, R_n$. For example, the image R in Fig. 9.1(a) is divided into three subregions R_1, R_2, and R_3 as shown in Figs 9.1(b) and 9.1(c). A subregion or sub-image is a portion of the whole region R. The identified subregions should exhibit characteristics such as uniformity and homogeneity with respect to colour, texture, intensity, or any other statistical property. In addition, the boundaries that separate the regions should be simple and clear.

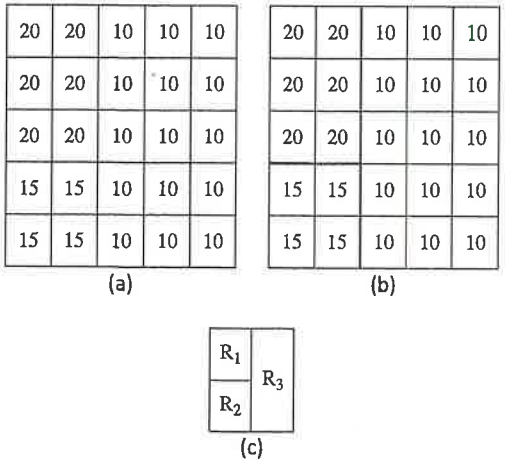


Fig. 9.1 Image segmentation (a) Original image (b) Pixels that form a region (c) Image with three regions

The characteristics of the segmentation process are the following:

1. If the subregions are combined, the original region can be obtained. Mathematically, it can be stated that $\bigcup R_i = R$ for $i = 1, 2, \ldots, n$. For example, if the three regions of Fig. 9.1(c) R_1, R_2, and R_3 are combined, the whole region R is obtained.
2. The subregions R_i should be connected. In other words, the region cannot be open-ended during the tracing process.
3. The regions $R_1, R_2, \ldots, R_n$ do not share any common property. Mathematically, it can be stated as $R_i \bigcap R_j = \varphi$ for all i and j where $i \neq j$. Otherwise, there is no justification for the region to exist separately.

4. Each region satisfies a predicate or a set of predicates such as intensity or other image statistics, that is, the predicate (P) can be colour, grey scale value, texture, or any other image statistic. Mathematically, this is stated as $P(R_i) = \text{True}$.

9.2 CLASSIFICATION OF IMAGE SEGMENTATION ALGORITHMS

There are different ways of classifying the segmentation algorithms. Figure 9.2 illustrates these ways. One way is to classify the algorithms based on user interaction required for extracting the ROI. Another way is to classify them based on the pixel relationships.

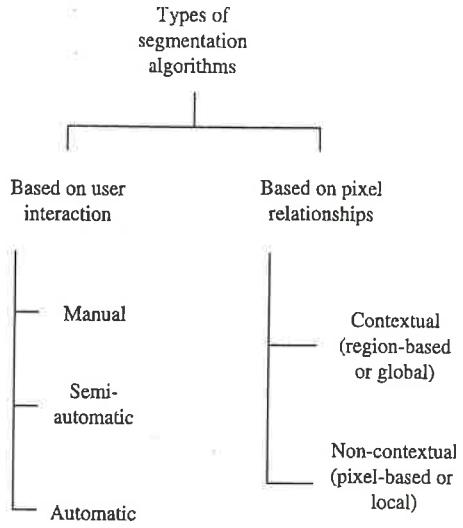


Fig. 9.2 Classification of segmentation algorithms

Based on user interaction, the segmentation algorithms can be classified into the following three categories:

1. Manual
2. Semi-automatic
3. Automatic

The words 'algorithm' and 'method' can be used interchangeably. In the manual method, the object of interest is observed by an expert who traces its ROI boundaries as well, with the help of software. Hence, the decisions related to segmentation are made by the human observers. Many software systems assist experts in tracing the boundaries and extracting them. By using the software systems, the experts outline the object.

The outline can be either an open or closed contour. Some software systems provide additional help by connecting the open tracings automatically to give a closed region. These closed outlines are then converted into a series of control points. These control points are then connected by spline. The advantage of the control points is that even if there is a displacement, the software system ensures that they are always connected. Finally, the software provides help to the user in extracting the closed regions.

Boundary tracing is a subjective process and hence variations exist among opinions of different experts in the field, leading to problems in reproducing the same results. In addition, a manual method of extraction is time consuming, highly subjective, prone to human error, and has poor intra-observer reproducibility. However, manual methods are still used commonly by experts to verify and validate the results of automatic segmentation algorithms.

Automatic segmentation algorithms are a preferred choice as they segment the structures of the objects without any human intervention. They are preferred if the tasks need to be carried out for a large number of images.

Semi-automatic algorithms are a combination of automatic and manual algorithms. In semi-automatic algorithms, human intervention is required in the initial stages. Normally, the human observer is supposed to provide the initial seed points indicating the ROI. Then the extraction process is carried out automatically as dictated by the logic of the segmentation algorithm. Region-growing techniques are semi-automatic algorithms where the initial seeds are given by the human observer in the region that needs to be segmented. However, the program process is automatic. These algorithms can be called assisted manual segmentation algorithms.

Another way of classifying the segmentation algorithms is to use the criterion of the pixel similarity relationships with neighbouring pixels. The similarity relationships can be based on colour, texture, brightness, or any other image statistic. On this basis, segmentation algorithms can be classified as follows:

1. Contextual (region-based or global) algorithms
2. Non-contextual (pixel-based or local) algorithms

Contextual algorithms group pixels together based on common properties by exploiting the relationships that exist among the pixels. These are also known as region-based or global algorithms. In region-based algorithms, the pixels are grouped based on some sort of similarity that exists between them. Non-contextual algorithms are also known as pixel-based or local algorithms. These algorithms ignore the relationship that exists between the pixels or features. Instead, the idea is to identify the discontinuities that are present in the image such as isolated lines and edges. These are then simply grouped into a region based on some global-level property. Intensity-based thresholding is a good example of this method.

9.3 DETECTION OF DISCONTINUITIES

The three basic types of grey level discontinuities in a digital image are the following:
1. Points 2. Lines 3. Edges

9.3.1 Point Detection

An isolated point is a point whose grey level is significantly different from its background in a homogeneous area. A generic 3 × 3 spatial mask is shown in Fig. 9.3.

The mask is superimposed onto an image and the convolution process is applied. The response of the mask is given as

$$R = \sum_{k=1}^{9} z_k f_k$$

where the f_k values are the grey level values of the pixels associated with the image. A threshold value T is used to identify the points. A point is said to be detected at the location on which the mask is centred if $|R| \geq T$, where T is a non-negative integer. The mask values of a point detection mask are shown in Fig. 9.4.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Fig. 9.3 Generic 3 × 3 spatial mask

1	1	1
1	−8	1
1	1	1

Fig. 9.4 Point detection mask

9.3.2 Line Detection

In line detection, four types of masks are used to get the responses, that is, R_1, R_2, R_3, and R_4 for the directions vertical, horizontal, +45°, and −45°, respectively. The masks are shown in Fig. 9.5(a).

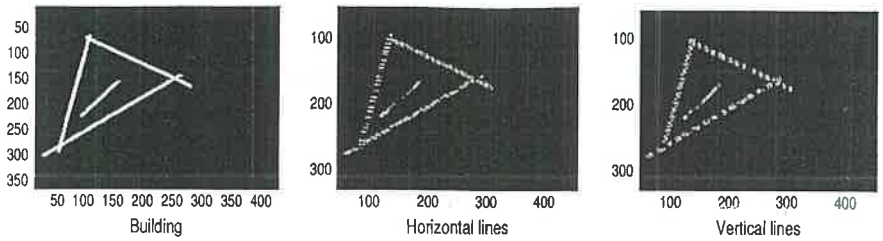
These masks are applied to the image. The response of the mask is given as $R_k = \sum_{k=1}^{4} z_k f_k$.

R_1 is the response for moving the mask from the left to the right of the image. R_2 is the response for moving the mask from the top to the bottom of the image. R_3 is the response of the mask along the +45° line and R_4 is the response of the mask with respect to a line of −45°. Suppose at a certain line on the image, $|R_i| > |R_j| \; \forall \; j \neq i$, then that line is more likely to be associated with the orientation of the mask. The final maximum response is defined by $\max_{i=1}^{4}\{R_i\}$ and the line is associated with that mask. A sample image and the results of the line-detection algorithm are shown in Fig. 9.5(b).

$$M_1 = \begin{pmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{pmatrix}, M_2 = \begin{pmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{pmatrix}, M_3 = \begin{pmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{pmatrix}, M_4 = \begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix}$$

(a)

Fig. 9.5 Line detection (a) Mask for line detection



(b)

Fig. 9.5 (b) Original image and detected lines

9.4 EDGE DETECTION

Edges play a very important role in many image processing applications. They provide an outline of the object. In the physical plane, edges correspond to the discontinuities in depth, surface orientation, change in material properties, and light variations. These variations are present in the image as grey scale discontinuities. An edge is a set of connected pixels that lies on the boundary between two regions that differ in grey value. The pixels on an edge are called edge points. A reasonable definition of an edge requires the ability to measure grey level transitions in a meaningful manner. Most edges are unique in space, that is, their position and orientation remain the same in space when viewed from different points. When an edge is detected, the unnecessary details are removed, while only the important structural information is retained. In short, an edge is a local concept which represents only significant intensity transitions. An original image and its edges are shown in Figs 9.6(a) and 9.6(b), respectively.

An edge is typically extracted by computing the derivative of the image function. This consists of two parts—magnitude of the derivative, which is an indication of the strength/contrast of the edge, and the direction of the derivative vector, which is a measure of edge orientation. Some of the edges that are normally encountered in image processing are as follows:

1. Step edge
2. Ramp edge
3. Spike edge
4. Roof edge

(a) (b)

Fig. 9.6 Edge detection (a) Original image (b) Extracted edges

These are shown in Fig. 9.7.

Step edge is an abrupt intensity change. Ramp edge, on the other hand, represents a gradual change in intensity. Spike edge represents a quick change and immediately returns to the original intensity level. Roof edge is not instantaneous over a short distance.

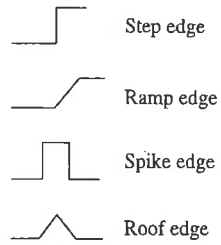


Fig. 9.7 Types of edges

9.4.1 Stages in Edge Detection

The idea is to detect the sharp changes in image brightness, which can capture the important events and properties. This is done in three stages. The edge detection process is shown in Fig. 9.8.

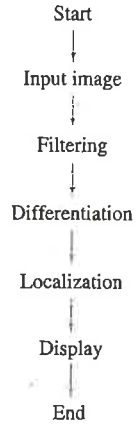


Fig. 9.8 Edge detection process

9.4.1.1 Filtering

It is better to filter the input image to get maximum performance for the edge detectors. This stage may be performed either explicitly or implicitly. It involves smoothing, where the noise is suppressed without affecting the true edges. In addition, this phase uses a filter to enhance the quality of the edges in the image. Normally, Gaussian filters are used as they are proven to be very effective for real-time images.

9.4.1.2 Differentiation

This phase distinguishes the edge pixels from other pixels. The idea of edge detection is to find the difference between two neighbourhood pixels. If the pixels have the same value, the difference is 0. This means that there is no transition between the pixels. The non-zero difference indicates the presence of an edge point. A point is defined as an edge point (or edgel) if its first derivative is greater than the user-specified threshold and encounters a sign change (zero crossing) in the second derivative. The first derivative is $\frac{\partial x}{\partial y} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x}$. Images are discrete. Hence, Δx should be discrete and should be at least 1. Therefore, the gradient vector ∇x (called grad of x) should be equal to $f(x) - f(x - 1)$. If the intensities are same, the derivative is 0. The non-zero element indicates the presence of images. In the case of the second derivatives, the zero-crossings indicate the presence of edges.

Example 9.1 Consider a one-dimensional image $f(x) = 60\ 60\ 60\ 100\ 100\ 100$. What are the first and second derivatives?

Solution The given one-dimensional image is

60 60 60 100 100 100

The first derivative is $f(x+1) - f(x)$. Therefore, the first derivative for the function is given as

0 0 40 0 0

The number 40 is due to the difference (100 – 60 = 40). Remaining values are all zeros. The second derivative is the difference in values of the first derivative. This is now given as

0 40 –40 0

The number 40 is due to the difference (40 – 0) and –40 is due to the difference (0 – 40). Remaining values are all zeros.

The highest magnitude 40 shows the presence of an edge in the first derivative. It can be observed that the sign change in the second derivative represents the edge. This sign change is important and is called zero-crossing.

Images are two-dimensional. Hence, the gradient vector of $f(x, y)$ is also two-dimensional. The gradient of an image $f(x, y)$ at location (x, y) is a vector that consists of the partial derivatives of $f(x, y)$ as follows:

$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix}$$

or simply

$$\nabla f(x,y) = \begin{bmatrix} g_x \\ g_y \end{bmatrix}$$

where $$g_x = \left[\frac{\partial f(x,y)}{\partial x}\right] \text{ and } g_y = \left[\frac{\partial f(x,y)}{\partial y}\right]$$

The magnitude of this vector, generally referred to as the gradient ∇f, is

$$\nabla f(x,y) = \text{mag}(\nabla f(x,y)) = \left[(g_x)^2 + (g_y)^2\right]^{1/2}$$

Edge strength is indicated by the edge magnitude. The direction of the gradient vector is useful in detecting a sudden change in image intensity. The common practice is to approximate the gradient with absolute values that are simpler to implement, as follows:

$$\nabla f(x,y) \cong |g_x| + |g_y|$$

or

$$\nabla f(x,y) \cong \max(g_x, g_y)$$

The gradient direction can be given as

$$\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$$

9.4.1.3 *Localization*

In this stage the detected edges are localized. The localization process involves determining the exact location of the edge. In addition, this stage involves edge thinning and edge linking steps to ensure that the edge is sharp and connected. The sharp and connected edges are then displayed.

The prerequisite for the localization stage is normalization of the gradient magnitude. The calculated gradient can be scaled to a specific range say, 0–K by performing this operation. For example, the value of constant K may be an integer, say, 100. $N(x, y)$ is called the normalized edge image and is given as

$$N(x,y) = \frac{G(x,y)}{\max_{i=1,\ldots,n,\, j=1,\ldots,n} G(i,j)} \times K$$

The normalized magnitude can be compared with a threshold value T to generate the edge map.

The edge map is given as

$$E(x,y) = \begin{cases} 1 \text{ if } N(x,y) > 1 \\ 0 \text{ otherwise} \end{cases}$$

The edge map is then displayed or stored for further image processing operations.

9.4.2 Types of Edge Detectors

The edge detection process is implemented in all kinds of edge detectors. In image processing, four types of edge detection operators are available. As shown in Fig. 9.9, they are gradient (or derivative) filters, template matching filters, Gaussian derivatives, and pattern fit approach.

Derivative filters use the differentiation technique to detect the edges. *Template matching filters* use templates that resemble the target shapes and match with the image. Gradient operations are isotropic in nature as they detect edges in all directions. Hence template matching filters are used to perform directional smoothing as they are very sensitive to directions. If there is a match between the target shape or directions and the masks, then a maximum gradient value is produced. By rotating the template in all eight directions, masks that are sensitive in all directions, called compass masks, are produced. Point detection and

line detection masks are good examples of template matching filters. *Gaussian derivatives* are very effective for real-time images and are used along with the derivative filters. *Pattern fit* is another approach, where a surface is considered as a topographic surface, with the pixel value representing altitude. The aim is to fit a pattern over a neighbourhood of a pixel where the edge strength is calculated. The properties of the edge points are calculated based on the parameters.

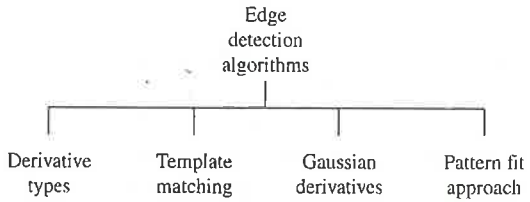


Fig. 9.9 Types of edge detectors

9.4.3 First-order Edge Detection Operators

Local transitions among different image intensities constitute an edge. Therefore, the aim is to measure the intensity gradients. Hence, edge detectors can be viewed as gradient calculators. Based on differential geometry and vector calculus, the gradient operator is represented as

$$\nabla = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix}$$

Applying this to the image f, one gets

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The differences between the pixels are quantified by the gradient magnitude. The direction of the greatest change is given by the gradient vector. This gives the directions of the edge. Since the gradient functions are continuous functions, the discrete versions of continuous functions can be used. This can be done by finding the differences. The approaches in 1D are as follows. Here Δx and Δy are the movements in x and y directions, respectively.

$$\text{Backward difference} = \frac{f(x) - f(x - \Delta x)}{\Delta x}$$

$$\text{Forward difference} = \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

$$\text{Central difference} = \frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x}$$

These differences can be obtained by applying the following masks, assuming $\Delta x = 1$:

$$\text{Backward difference} = f(x) - f(x - 1) = [1 \; -1]$$

$$\text{Forward difference} = f(x + 1) - f(x) = [-1 \; +1]$$

$$\text{Central difference} = 1/2 \times [1 \; 0 \; -1]$$

These differences can be extended to 2D as $g_x = \frac{\partial f}{\partial x}$ and $g_y = \frac{\partial f}{\partial y}$. Then the magnitude is given by

$$\nabla f(x, y) = \text{mag}(\nabla f(x, y)) = \left[(g_x)^2 + (g_y)^2 \right]^{1/2}$$

The gradient direction is given by $\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$.

9.4.3.1 Roberts operator

Let $f(x,y)$ and $f(x + 1, y)$ be neighbouring pixels. The difference between the adjacent pixels is obtained by applying the mask [1 −1] directly to the image to get the difference between the pixels. This is defined mathematically as

$$\frac{\partial f}{\partial x} = f(x + 1, y) - f(x, y)$$

Roberts kernels are derivatives with respect to the diagonal elements. Hence they are called cross-gradient operators. They are based on the cross diagonal differences. The approximation of Roberts operator can be mathematically given as

$$g_x = \frac{\partial f}{\partial x} = (z_9 - z_5)$$

$$g_y = \frac{\partial f}{\partial y} = (z_8 - z_6)$$

Roberts masks of the for the given cross difference is

$$g_x = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \text{ and } g_y = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

As discussed earlier the magnitude of this vector can be calculated as

$$\nabla f(x,y) = \text{mag}(\nabla f(x,y)) = \left[(g_x)^2 + (g_y)^2\right]^{1/2}$$

and the edge orientation is given by

$$\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$$

Since the magnitude calculation involves square root operation, the common practice is to approximate the gradient with absolute values that are simpler to implement, as

$$\nabla f(x,y) \cong |g_x| + |g_y|$$

The generic gradient-based algorithm can be given as

1. Read the image and smooth it.
2. Convolve the image f with g_x. Let $\hat{f}(x) = f * g_x$.
3. Convolve the image with g_y. Let $\hat{f}(y) = f * g_y$.
4. Compute the edge magnitude and edge orientation.
5. Compare the edge magnitude with a threshold value. If the edge magnitude is higher, assign it as a possible edge point.

This generic algorithm can be applied to other masks also.

9.4.3.2 Prewitt operator

The Prewitt method takes the central difference of the neighbouring pixels; this difference can be represented mathematically as

$$\frac{\partial f}{\partial x} = f(x+1) - f(x-1)/2$$

For two dimensions, this is

$$f(x+1, y) - f(x-1, y)/2$$

The central difference can be obtained using the mask [−1 0 +1]. This method is very sensitive to noise. Hence to avoid noise, the Prewitt method does some averaging. The Prewitt approximation using a 3×3 mask is as follows:

$$\nabla f \cong |(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)| + |(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)|$$

This approximation is known as the Prewitt operator. Its masks are as follows:

$$M_x = \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \text{ and } M_y = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 0 \\ -1 & 0 & 1 \end{pmatrix}$$

9.4.3.3 Sobel operator

The Sobel operator also relies on central differences. This can be viewed as an approximation of the first Gaussian derivative. This is equivalent to the first derivative of the Gaussian blurring image obtained by applying a 3 × 3 mask to the image. Convolution is both commutative and associative, and is given as

$$\frac{\partial}{\partial x}(f * G) = f * \frac{\partial}{\partial x} G$$

A 3 × 3 digital approximation of the Sobel operator is given as

$$\nabla f \cong |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|$$

The masks are as follows:

$$M_x = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \text{ and } M_y = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

An additional mask can be used to detect the edges in the diagonal direction.

$$M_x = \begin{pmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{pmatrix} \text{ and } M_y = \begin{pmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix}$$

The edge mask can be extended to 5 × 5, 7 × 7, etc. An extended mask always gives a better performance. An original image and the result of applying the Roberts, Sobel, and Prewitt masks are shown in Figs 9.10(a)–9.10(d). It can be observed that the results of the masks vary.

(a) (b) (c) (d)

Fig. 9.10 Edge detection using first-order operators (a) Original image (b) Roberts edge detection (b) Prewitts edge detection (c) Sobel edge detection

9.4.3.4 Template matching masks

Gradient masks are isotropic and insensitive to directions. Sometimes it is necessary to design direction sensitive filters. Such filters are called *template matching filters*. A few kinds of template matching masks are discussed in this section.

Kirsch masks Kirsch masks are called compass masks because they are obtained by taking one mask and rotating it to the eight major directions: north, north west, west, south west, south, south-east, east, and north-east. The respective masks are given as

$$K_0=\begin{bmatrix}-3 & -3 & 5\\ -3 & 0 & 5\\ -3 & -3 & 5\end{bmatrix} K_1=\begin{bmatrix}-3 & 5 & 5\\ -3 & 0 & 5\\ -3 & -3 & -3\end{bmatrix} K_2=\begin{bmatrix}5 & 5 & 5\\ -3 & 0 & -3\\ -3 & -3 & -3\end{bmatrix} K_3=\begin{bmatrix}5 & 5 & -3\\ 5 & 0 & -3\\ -3 & -3 & -3\end{bmatrix}$$

$$K_4=\begin{bmatrix}5 & -3 & -3\\ 5 & 0 & -3\\ 5 & -3 & -3\end{bmatrix} K_5=\begin{bmatrix}-3 & -3 & -3\\ 5 & 0 & -3\\ 5 & 5 & -3\end{bmatrix} K_6=\begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & -3\\ 5 & 5 & 5\end{bmatrix} K_7=\begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & 5\\ -3 & 5 & 5\end{bmatrix}$$

Each mask is applied to the image and the convolution process is carried out. The magnitude of the final edge is the maximum value of all the eight masks. The edge direction is the direction associated with the mask that produces maximum magnitude.

Robinson compass mask The spatial masks for the Robinson edge operator for all the directions are as follows:

$$R_0=\begin{pmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{pmatrix} R_1=\begin{pmatrix}0 & 1 & 2\\ -1 & 0 & 1\\ -2 & -1 & 0\end{pmatrix} R_2=\begin{pmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{pmatrix},$$

$$R_3=\begin{pmatrix}2 & 1 & 0\\ 1 & 0 & -1\\ 0 & -1 & -2\end{pmatrix} R_4=\begin{pmatrix}1 & 0 & -1\\ 2 & 0 & -2\\ 1 & 0 & -1\end{pmatrix} R_5=\begin{pmatrix}0 & -1 & -2\\ 1 & 0 & -1\\ 2 & 1 & 0\end{pmatrix},$$

$$R_6=\begin{pmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{pmatrix} R_7=\begin{pmatrix}-2 & -1 & 0\\ -1 & 0 & 1\\ 0 & 1 & 2\end{pmatrix}$$

Similar to Kirsch masks, the mask that produces the maximum value defines the direction of the edge. It is sufficient for edge detection. The results of the remaining masks are the negation of the first four masks. Thus the computational effort can be reduced.

Frei-Chen masks Any image can be considered as the weighted sum of the nine Frei-Chen masks. The weights are obtained by a process called projecting process by overlaying a 3 × 3 image onto each mask and by summing the multiplication of coincident terms. The first four masks represent the edge space, the next four represent the line subspace, and the last one represents the average subspace. The Frei-Chen masks are given as

$$F_1=\frac{1}{2\sqrt{2}}\begin{pmatrix}1 & \sqrt{2} & 1\\ 0 & 0 & 0\\ -1 & -\sqrt{2} & -1\end{pmatrix} \quad F_2=\frac{1}{2\sqrt{2}}\begin{pmatrix}1 & 0 & -1\\ \sqrt{2} & 0 & -\sqrt{2}\\ 1 & 0 & -1\end{pmatrix}$$

$$F_3=\frac{1}{2\sqrt{2}}\begin{pmatrix}0 & -1 & \sqrt{2}\\ 1 & 0 & -1\\ \sqrt{2} & 1 & 0\end{pmatrix} \quad F_4=\frac{1}{2\sqrt{2}}\begin{pmatrix}\sqrt{2} & -1 & 0\\ -1 & 0 & 1\\ 0 & 1 & \sqrt{2}\end{pmatrix}$$

$$F_5=\frac{1}{2}\begin{pmatrix}0 & 1 & 0\\ -1 & 0 & -1\\ 0 & 1 & 0\end{pmatrix} \quad F_6=\frac{1}{2}\begin{pmatrix}-1 & 0 & -1\\ 0 & 0 & 0\\ 1 & 0 & -1\end{pmatrix}$$

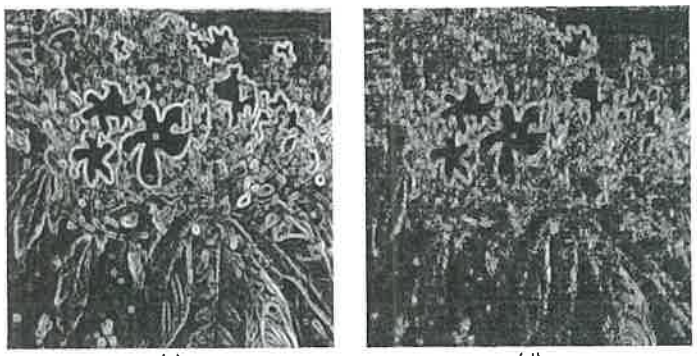
$$F_7=\frac{1}{6}\begin{pmatrix}1 & -2 & 1\\ -2 & 4 & -2\\ 1 & -2 & 1\end{pmatrix} \quad F_8=\frac{1}{6}\begin{pmatrix}-2 & 1 & -2\\ 1 & 4 & 1\\ -2 & 1 & -2\end{pmatrix}$$

$$F_9=\frac{1}{3}\begin{pmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{pmatrix}$$

Figures 9.11(a)–9.11(d) show an original image and the images obtained by using the Kirsch, Robinson compass, and Frei-Chen masks, respectively.

(a) (b)

Fig. 9.11 Template matching masks (a) Original image (b) Image obtained using Krisch mask



(c) (d)

Fig. 9.11 (c) Image obtained using Robinson compass mask
(d) Image obtained using Frei-Chen mask

9.4.4 Second-order Derivative Filters

Edges are considered to be present in the first derivative when the edge magnitude is large compared to the threshold value. In the case of the second derivative, the edge pixel is present at a location where the second derivative is zero. This is equivalent to saying that $f''(x)$ has a zero-crossing which can be observed as a sign change in pixel differences. The Laplacian algorithm is one such zero-crossing algorithm.

However, the problems of the zero-crossing algorithms are many. The problem with Laplacian masks is that they are sensitive to noise as there is no magnitude checking—even a small ripple causes the method to generate an edge point. Therefore, it is necessary to filter the image before the edge detection process is applied. This method produces two-pixel thick edges, although generally, one-pixel thick edges are preferred. However, the advantage is that there is no need for the edge thinning process as the zero-crossings themselves specify the location of the edge points. The main advantage is that these operators are rotationally invariant.

The second-order derivative is

$$\nabla \times \nabla = \begin{bmatrix} \partial/\partial x \\ \partial/\partial y \end{bmatrix} \begin{bmatrix} \partial/\partial \\ \partial/\partial y \end{bmatrix} = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

This ∇^2 operator is called Laplacian operator. The Laplacian of the 2D function $f(x, y)$ is also defined as

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

Since the gradient is a vector, two orthogonal filters are required. However, since the Laplacian operator is scalar, a single mask is sufficient for the edge detection process. The Laplacian estimate is given as

$$\begin{aligned} &\frac{\partial^2 f(x,y)}{\partial y^2} \\ &= \frac{\delta}{\delta x} f(x+1,y) - \frac{\delta}{\delta x} f(x,y) \\ &= f(x+1,y) - f(x,y) - [f(x,y) - f(x-1,y)] \\ &= f(x+1,y) - f(x,y) - f(x,y) + f(x-1,y) \end{aligned}$$

Similarly, $\frac{\partial^2 f(x,y)}{\partial y^2} = f(x, y+1) - 2f(x, y) + f(x, y-1)$

Therefore, the Laplacian operator has the form

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} = f(x+1, y) + f(x, y+1) - 4f(x, y) + f(x, y-1) + f(x-1, y)$$

The algorithm is as follows:

1. Generate the mask.
2. Apply the mask.
3. Detect the zero-crossing. Zero-crossing is a situation where pixels in a neighbourhood differ from each other pixel in sign, that is, $|\nabla^2 f(p)| \leq |\nabla^2 f(q)|$, where p and q are two pixels.

Laplacian masks are shown in Figs 9.12(a)–9.12(d). The mask shown in Fig. 9.12(a) is sensitive to horizontal and vertical edges. It can be observed that the sum of the elements amounts to zero. To recognize the diagonal edges, the mask shown in Fig. 9.12(b) is used. This mask is obtained by rotating the mask of Fig. 9.12(a) by 45°. The addition of these two kernels results in a variant of the Laplacian mask shown in Fig. 9.12(c). Two times of the mask shown in Fig. 9.12(a), when subtracted from the mask shown in Fig. 9.12(b), yields another variant mask as shown in Fig. 9.12(d).

0	1	0
1	−4	1
0	1	0

(a)

1	0	1
0	−4	0
1	0	1

(b)

1	1	1
1	−8	1
1	1	1

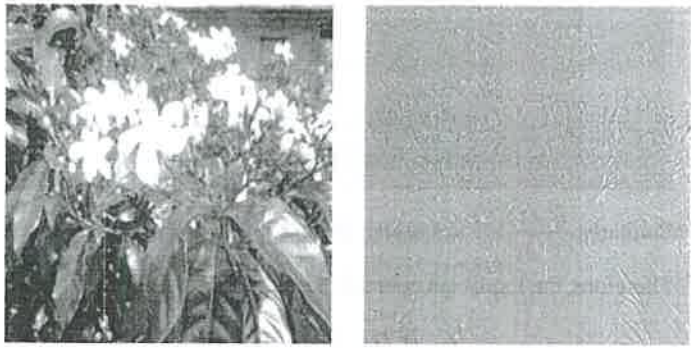
(c)

1	−2	1
−2	4	−2
1	−2	1

(d)

Fig. 9.12 Different Laplacian masks (a) Laplacian filter
(b) 45° rotated mask (c) Variant 1 (d) Variant 2

The Laplacian operations are seldom used in practice because they produce double edges and are extremely sensitive to noise. However, the idea of zero-crossing is useful if it is combined with a smoothing signal to minimize sensitivity to noise. The result of applying the Laplacian method on Fig. 9.13(a) is shown in Fig. 9.13(b).



(a) (b)

Fig. 9.13 Laplacian method (a) Original image (b) Result of applying Laplacian mask

9.4.4.1 Laplacian of Gaussian (Marr–Hildrith) operator

To minimize the noise susceptibility of the Laplacian operator, the Laplacian of Gaussian (LoG) operator is often preferred. As a first step, the given image is blurred using the Gaussian operator and then the Laplacian operator is used. The Gaussian function reduces the noise and hence the Laplacian minimizes the detection of false edges.

For 1D, $\nabla^2(f * g) = f * \nabla^2 g = f * \text{LoG}$. Let the 2D Gaussian function be given as $G_\sigma(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{x^2+y^2}{2\sigma^2})$. To suppress the noise, the image is convolved with the Gaussian smoothing function before using the Laplacian for edge detection.

$$\nabla(G_\sigma(x,y) * f(x,y)] = [\nabla G_\sigma(x,y)] * f(x,y) = \text{LoG} * f(x,y)$$

The LoG function can be derived as

$$\frac{\partial}{\partial x} G_\sigma(x,y) = \frac{\partial}{\partial x} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)} = -\frac{x}{\sigma^2} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}$$

Similarly $\frac{\partial^2}{\partial x^2} G_\sigma(x,y) = \frac{x^2}{\sigma^4} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)} - \frac{1}{\sigma^2} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)} = \frac{x^2-\sigma^2}{\sigma^4} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}$

Similarly, by ignoring the normalization constant $\frac{1}{\sqrt{2\pi\sigma^2}}$, one can get

$$\frac{\partial^2}{\partial y^2} G_\sigma(x,y) = \frac{y^2}{\sigma^4} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)} - \frac{1}{\sigma^2} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)} = \frac{y^2-\sigma^2}{\sigma^4} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}$$

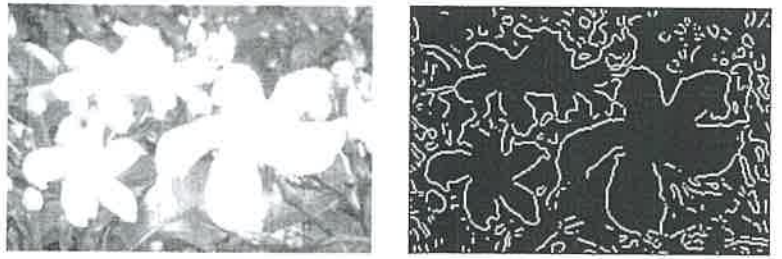
The LoG kernel can now be described as

$$LoG \triangleq \frac{\partial^2}{\partial x^2} G_\sigma(x,y) + \frac{\partial^2}{\partial y^2} G_\sigma(x,y) = \frac{x^2+y^2-2\sigma^2}{\sigma^4} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}$$

As σ increases, wider convolution masks are required for better performance of the edge operator. A sample of an image and its result after application of the LoG operator are shown in Figs 9.14(a) and 9.14(b), respectively.

The LoG algorithm can now be stated as follows:

1. Generate the mask and apply LoG to the image.
2. Detect the zero-crossing.



(a) (b)

Fig. 9.14 Laplacian of Gaussian operator (a) Original image (b) Result of applying LoG operator

9.4.4.2 Combined detection

The information obtained from different orthogonal operators can be combined to get better results. One method is to combine the first- and second-order derivatives. This can be achieved by implementing the idea of scale space. Just like how the information of an object can be captured at different levels using different apertures of a camera, different types of Gaussian kernels of various sigma values can be used to capture information of the image at different levels and these information can be combined to get the edge map.

Another method called *hysteresis thresholding* uses two thresholds for thresholding the image. This is used by the canny edge detector method.

9.4.4.3 Difference of Gaussian filter

The LoG filter can be approximated by taking two differently sized Gaussians. The Difference of Gaussian (DoG) filter is given as

$$G_{\sigma_1}(x,y)=\frac{1}{\sqrt{2\pi\sigma_1^2}}\exp\left(-\frac{x^2+y^2}{2\sigma_1^2}\right) \text{ with Gaussian of width } \sigma_1$$

The width of the Gaussian is changed and a new kernel is obtained as

$$G_{\sigma_2}(x,y)=\frac{1}{\sqrt{2\pi\sigma_2^2}}\exp\left(-\frac{x^2+y^2}{2\sigma_2^2}\right) \text{ with Gaussian of width } \sigma_2$$

The DoG is expressed as the difference between these two Gaussian kernels:

$$G_{\sigma_1}(x,y) - G_{\sigma_2}(x,y) = (G_{\sigma_1} - G_{\sigma_2}) * f(x,y)$$
$$= \text{DoG} * f(x,y)$$

The DoG as a kernel can now be stated as

$$\text{DoG} = G_{\sigma_1} - G_{\sigma_2} = \frac{1}{\sqrt{2\pi}}\left[\frac{1}{\sigma_1}e^{-\frac{(x^2+y^2)}{2\sigma_1^2}} - \frac{1}{\sigma_2}e^{-\frac{(x^2+y^2)}{2\sigma_2^2}}\right]$$

So the given image has to be convolved with a mask that is obtained by subtracting two Gaussian masks with two different σ values. If the $\frac{\sigma_1}{\sigma_2}$ value is between 1 and 2, the edge detection operator yields a good performance.

The DoG algorithm can now be stated as follows:

1. Generate the mask and apply DoG to the image.
2. Detect the zero-crossing and apply the threshold to suppress the weak zero-crossings.
3. Display and exit.

9.4.4.4 Canny edge detection

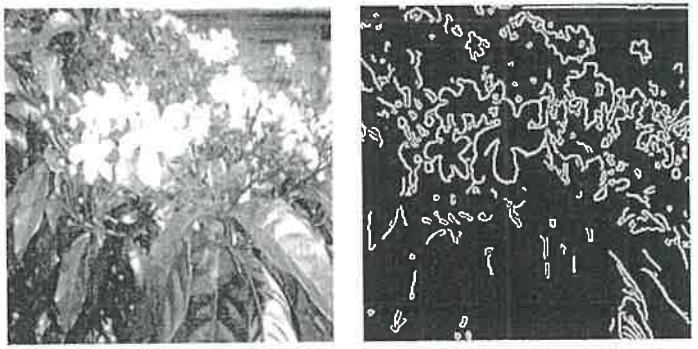
The Canny approach is based on optimizing the trade-off between two performance criteria and can be described as follows:

1. Good edge detection—The algorithm should detect only the real edge points and discard all false edge points.
2. Good edge localization—The algorithm should have the ability to produce edge points that are closer to the real edges.
3. Only one response to each edge—The algorithm should not produce any false, double, or spurious edges.

The Canny edge detection algorithm is given as follows:

1. First convolve the image with the Gaussian filter. Compute the gradient of the resultant smooth image. Store the edge magnitude and edge orientation separately in two arrays, $M(x, y)$ and $\alpha(x, y)$, respectively.

2. The next step is to thin the edges. This is done using a process called *non-maxima suppression*. Examining every edge point orientation is a computationally intensive task. To avoid such intense computations, the gradient direction is reduced to just four sectors. How? The range of 0–360° is divided into eight equal portions. Two equal portions are designated as one sector. Therefore there will be four sectors. The gradient direction of the edge point is first approximated to one of these sectors. After the sector is finalized, let us assume a point of $M(x, y)$. The edge magnitudes $M(x_1, y_1)$ and $M(x_2, y_2)$, of two neighbouring pixels that fall on the same gradient direction, are considered. If the magnitude of the point $M(x, y)$ is less than the magnitude of the points (x_1, y_1) or (x_2, y_2), then the value is suppressed, that is, the value is set to zero; otherwise the value is retained.
3. Apply hysteresis thresholding. The idea behind hysteresis thresholding is that only a large amount of change in the gradient magnitude matters in edge detection and small changes do not affect the quality of edge detection. This method uses two thresholds, t_0 and t_1. If the gradient magnitude is greater than the value t_1, it is considered as a definite edge point and is accepted. Similarly, if the gradient magnitude is less than t_0, it is considered as a weak edge point and removed. However, if the edge gradient is between t_0 and t_1, it is considered as either weak or strong based on the context. This is implemented by creating two images using two thresholds t_0 and t_1. Low threshold creates a situation where noisier edge points are accepted. On the other hand, a high value of the threshold removes many potential edge points. So this process first thresholds the image with low and high thresholds to create two separate images. The image containing the high threshold image will contain edges, but gaps will be present. So the image created using the low threshold is consulted and its 8-neighbours are examined. So the gaps of the high threshold image are bridged using the edge points of the low threshold image. This process thus ensures that the edges are linked properly to generate a perfect contour of the image. The results of the Canny detector are shown in Figs 9.15(a)–9.15(d).



(a) (b)

Fig. 9.15 Canny edge detection (a) Original image (b) Canny edge detection at $\sigma = 1$