CHAPTER 5

Image Enhancement

Contrast is what makes photography interesting.

–Conrad Hall

LEARNING OBJECTIVES

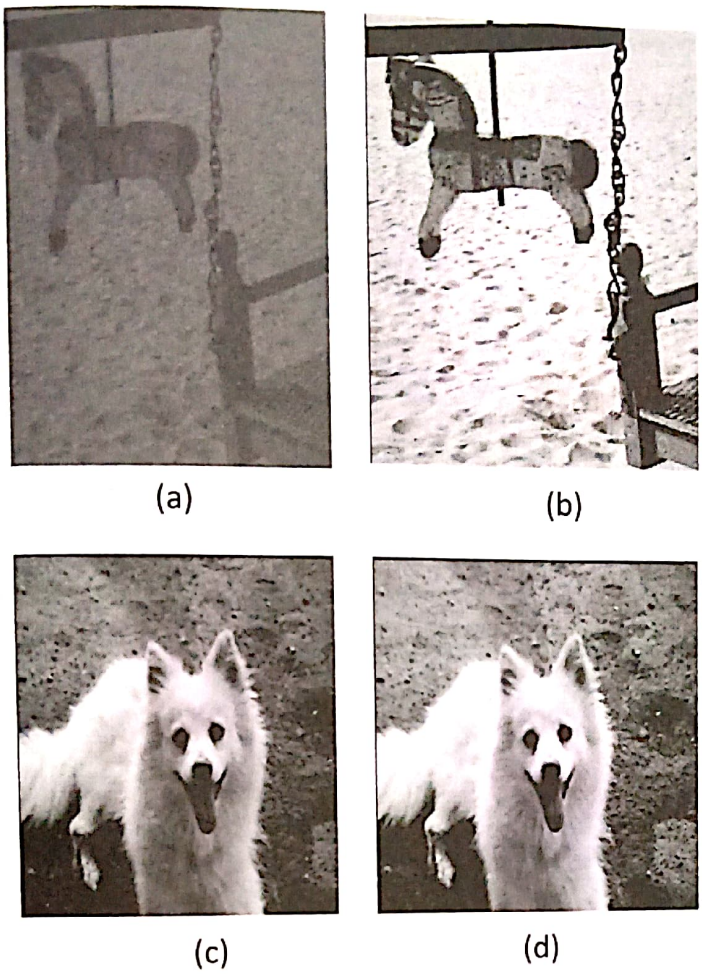
This chapter provides an overview of digital image enhancement. Image enhancement is the process of enhancing the quality of a given image for analysis. The aim of this process is to improve the quality of the image so that the image analysis is accurate, leading to an improvement in the reliability of the image processing applications. This chapter introduces concepts related to grey-level transformations and the concept of filtering, including spatial and frequency domain filtering. After studying this chapter, the reader will become familiar with the following:

- Basics of image quality
- Role of histogram in image quality assessment
- Grey-level transformations
- Spatial filters
- Frequency domain filtering

5.1 IMAGE QUALITY AND NEED FOR IMAGE ENHANCEMENT

An image is supposed to provide information to a viewer. For example, a medical image is supposed to provide information on specific parts of the body for diagnosis and decision-making. Similarly, a forensic image is supposed to provide accurate details to assist in forensic investigations. It is often necessary to extract various features of an image or the objects that are present in the image. Often the quality of the acquired images is not satisfactory due to factors such as brightness, contrast, blur, unnatural colours, noise, and artefacts.

Figures 5.1(a) and (c) show a dark image and a poor contrast image, respectively, in which the image details are not clear. Figure 5.1(b) is a relatively good image with enhanced brightness, and Fig. 5.1(d) is a much better contrast-enhanced image. These images demonstrate that it is necessary to pre-process an image so that irrelevant information or noise is removed. This process is called image enhancement.

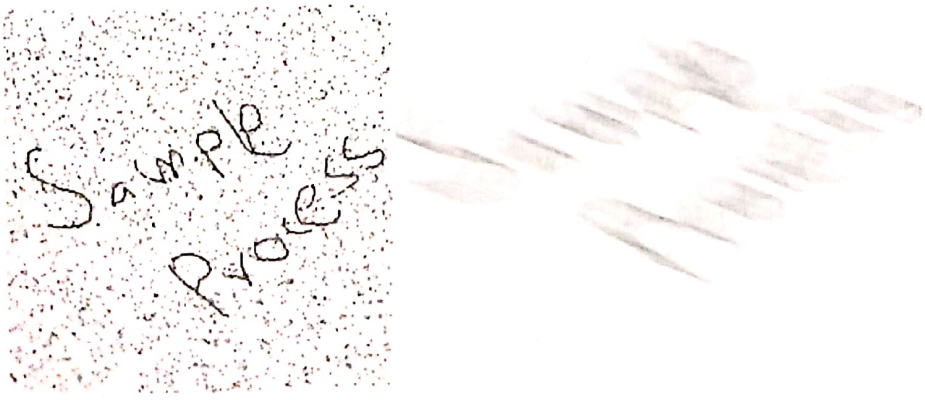


(a) (b)

(c) (d)

Fig. 5.1 Image enhancement problems (a) Dark image (b) Brightness-enhanced image (c) Poor contrast image (d) Contrast-enhanced image

Image enhancement conditions the input image so as to improve the quality of the image for further image processing operations, whereas image restoration is a complex process that is related to problems such as noise, blur, and artefacts. Figure 5.2(a) shows a noisy image and Fig. 5.2(b) shows a blurred image. These images are not useful because it is difficult to interpret the contents of the image due to noise and blur.



(a) (b)

Fig. 5.2 Image restoration problems (a) Image with noise (b) Image with blur

The differences between image enhancement and image restoration are listed in Table 5.1.

Table 5.1 Differences between image enhancement and image restoration

Image enhancement	Image restoration
Image enhancement is a subjective process.	Image restoration is an objective process and is based on sound mathematical principles.
It involves only cosmetic changes in the brightness and contrast.	It requires modelling of the degradations.
Image enhancement algorithms are heuristic in nature and often involve trial-and-error processes.	Image restoration algorithms are well defined.
The procedure is very simple.	The procedure is complex.

To improve the quality of an image, it is necessary to assess its quality. This requires some quality quantification and assessment tools. Some of the image quality factors are discussed in the next section.

5.1.1 Image Quality Factors

The discussion of image enhancement starts with the concept of image quality. Many factors are responsible for image quality; some of the essential factors are as follows:

1. Contrast
2. Brightness
3. Spatial resolution
4. Noise

Contrast This refers to the finer details of an image. All objects do not have the same kind of surface. Some surfaces of objects are smooth and some are rough. In fact, light itself does not illuminate the scene uniformly as some portions of an object may hide other portions of the same object or other objects partially. Similarly, objects reflect light in a non-uniform manner as their properties differ. Therefore, when an image is formed, the magnitude of the intensity differences between the different surfaces of an object is recorded. This is called contrast, and can be described as the product of sensor signal contrast (this depends on the energy source and the physical properties of the object) and detector contrast (this depends on the way a signal is detected, captured, and stored).

Contrast can be measured in many ways. A common measure of contrast involving foreground and background objects is given as follows:

$$C = \frac{f_{\text{object}} - f_{\text{background}}}{f_{\text{object}} + f_{\text{background}}}$$

where f_{object} is the average pixel intensity of the object and $f_{\text{background}}$ the average pixel intensity of the background.

Another expression used for calculating contrast is as follows:

$$C_{\text{Webber}} = \frac{f_{\text{object}} - f_{\text{background}}}{f_{\text{background}}}$$

This is often referred to as simultaneous contrast. A further way to compute contrast on sinusoidal grating is to calculate the difference between its peak and trough, which is given as follows:

$$C_{\text{Michelson}} = \frac{f_{\max} - f_{\min}}{f_{\max} + f_{\min}}$$

Here, $f_{\max}$ and $f_{\min}$ are the maximum and minimum intensities of the image.

Another useful contrast measurement using the same parameters related to foreground and background object is as follows:

$$C = \log_{10} \frac{f_{\text{object}}}{f_{\text{background}}}$$

Contrast can be increased by multiplying every pixel of the image $f(x, y)$ by a constant a, where the value of a is greater than 1. It makes the brighter samples even brighter and darker samples even darker. The new image obtained, $g(x, y)$, can then mathematically be expressed as follows:

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

On the other hand, if the value of a is less than 1, then the opposite effect takes place where the range is reduced. An original image along with the contrast-enhanced (obtained by multiplying by a constant of 1.5) and contrast-suppressed (obtained by multiplying by 0.5) images are shown in Figs 5.3(a)–(c).

(a) (b) (c)

Fig. 5.3 Contrast manipulation (a) Original image (b) Contrast-enhanced image (multiplying by 1.5) (c) Contrast-suppressed image (multiplying by 0.5)

Brightness This refers to the average pixel intensity of an image. Similar to contrast, the image should be reasonably bright to show all the information to the viewer. However, excessive brightness may affect the quality of the image. An image may have higher

brightness, but if the intensity is not optimal, the image will not be of good quality. Brightness resolution is defined as the number of grey levels in a monochrome image. In the case of a colour image, the number of distinct colours is called colour resolution.

The brightness can be increased by adding a constant b to every pixel of the image $f(x, y)$. The new image obtained, $g(x, y)$, can then mathematically be expressed as follows:

$$g(x, y) = f(x, y) + b$$

Similarly, brightness can be decreased by subtracting a constant b from every pixel of the image $f(x, y)$. The new image obtained, $g(x, y)$, can then mathematically be written as follows:

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

An original image, its brightness-enhanced (obtained by adding a constant 20) and brightness-suppressed (obtained by subtracting a constant 50) forms are shown in Figs 5.4(a)–(c).

(a) (b) (c)

Fig. 5.4 Brightness manipulation (a) Original image (b) Brightness-enhanced image (plus 20) (c) Brightness-suppressed image (minus 50)

Spatial resolution This has already been discussed in Chapter 2, deals with the number of pixels, pixel density, and quantization levels. This depends on the sampling and quantization processes. If spatial resolution is not satisfactory, the quality of an image will not be satisfactory. Generally, the quality of an image is supposed to be higher if it has more pixels. Therefore, usually, spatial resolution is defined as the number of pixels per inch of an image. However, this is not correct. An image may have more pixels, but it may not provide sufficient information regarding other factors like optical resolution. The distance of an object from the camera and the field of view also plays an important role in defining the image quality.

Image applications are frequently affected by the *noise* present in the image, which is an unwanted disturbance that causes fluctuations in the pixel value. It is a random or

stochastic process, and hence its true value cannot be predicted accurately. However, noise obeys all the statistical properties. Therefore, a pixel is characterized as a random variable for statistical analysis. Noise modelling will be discussed in Chapter 6.

5.1.2 Image Quality Assessment Tool

A histogram gives a useful summary of the distribution of grey levels in an image, by plotting the frequency of occurrence of the various grey levels. This plot provides a global description of the appearance of the image. The x- and y-axis of the histogram represent the grey level values and the number of pixels having that intensity, respectively.

A plot of the histogram for the image in Fig. 5.5(a) is shown in Fig. 5.5(b).

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

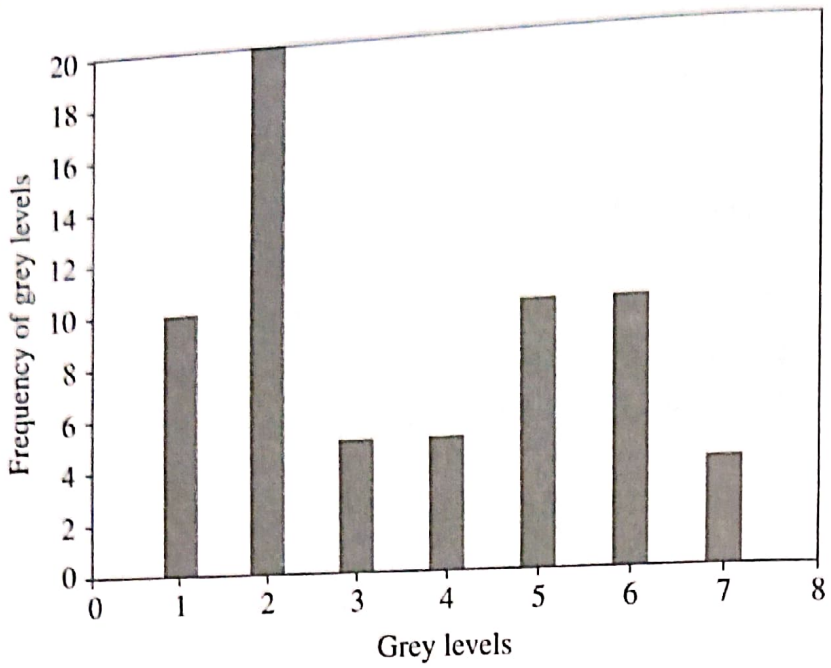
(a)

Fig. 5.5 Histogram as assessment tool (a) Sample image

Table 5.2 shows the frequency distribution of grey levels in a sample image.

Table 5.2 Frequency distribution of grey levels

Pixels	Frequency
0	0
1	10
2	20
3	5
4	5
5	10
6	10
7	4
Total	64



(b)

Fig. 5.5 (b) Histogram based on frequency distribution of grey levels

Another way of plotting a histogram is to use the probability of occurrence of the pixels.

5.1.2.1 Advantages of Histograms

The histogram of an image indicates the dynamic range of the image, that is, how well the image is spread out. Dynamic range is an indicator of contrast and of how good the image is. A sample image and its histogram are shown in Figs 5.6(a) and (b), respectively. In general, an image with a histogram of small spread has low contrast and one with a wide spread has high contrast. Dynamic range is a useful metric that is expressed as the difference between the maximum and minimum pixel values found in the histogram.

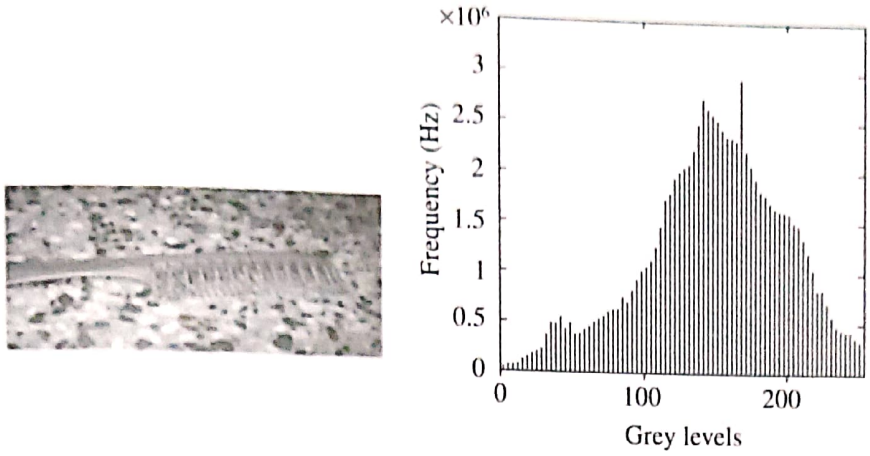
$$\text{Dynamic range} = f_{\max} - f_{\min}$$

Here, $f_{\max}$ is the maximum pixel value and $f_{\min}$ is the minimum pixel value of the image. The dynamic range can also be expressed as follows:

$$\text{Dynamic range} = 20 \log (f_{\max} - f_{\min}) \text{ dB (in decibels)}$$

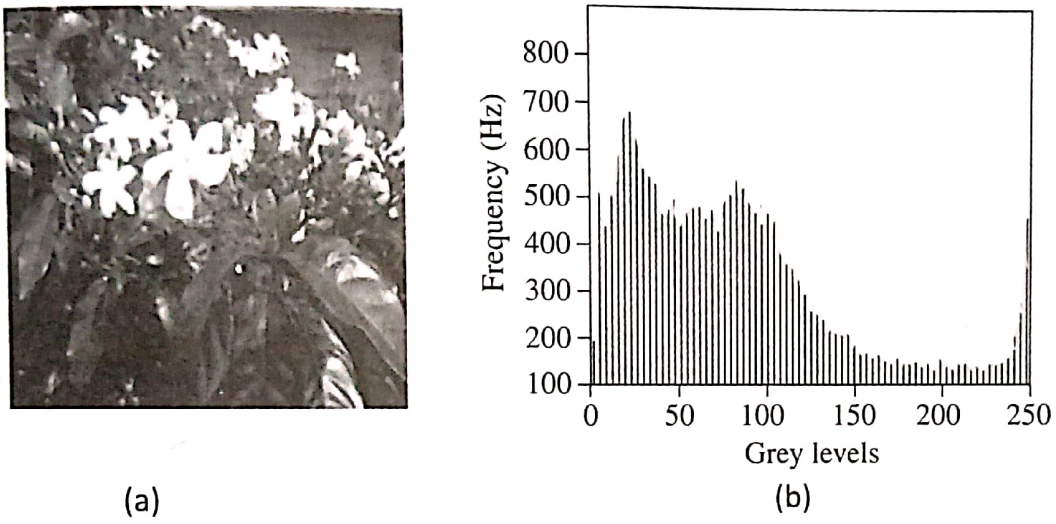
A histogram with values clustered at the low end of the range represents a dark image. Similarly, a histogram with values clustered at the high end of the range corresponds to a bright image.

Figures 5.7(a)–(d) illustrate the effect of image quality on a histogram. Figure 5.7(a) shows an original image, and its associated histogram is shown in Fig. 5.7(b). Figure 5.7(c) illustrates the effect of subtracting a factor of 100 from the original image, which results in a decrease of brightness. The histogram of a dark image is shown in Fig. 5.7(d). It can be observed that the lack of brightness is reflected as the lack of any significant peaks and valleys, unlike the histogram shown in Fig. 5.7(b).

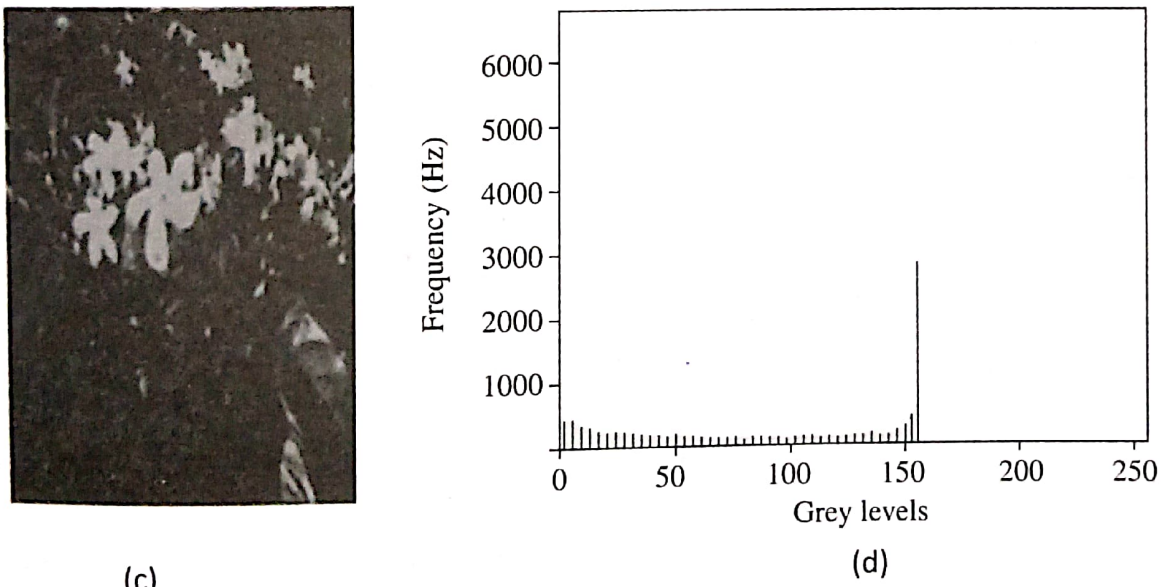


(a) (b)

Fig. 5.6 Illustration of a histogram (a) Sample image (b) Histogram of the sample image



(a) (b)



(c) (d)

Fig. 5.7 Effect of image quality on a histogram (a) Original image (b) Histogram of original image (c) Dark image (d) Histogram of dark image

5.1.3 Image Quality Metrics

It is necessary to quantify the quality of an image. The metrics used to quantify the image quality can be divided into the following two categories:

1. Objective fidelity criteria
2. Subjective fidelity criteria

Objective fidelity criteria provide equations that help us quantify the error, helping in characterizing image quality. Let us assume that the input image is $f(x, y)$. Suppose an image task, for example, image compression, is carried out, and the output of the process is another image—let us call it $\hat{f}(x, y)$. This means that the reference or ideal image is $f(x, y)$, and the image obtained after an image processing operation is $\hat{f}(x, y)$. The error can now be denoted as follows:

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

The problem with an error is that it may also be negative. To avoid negative numbers, the mean square error (MSE) is commonly used and can be expressed as follows:

$$\text{MSE} = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [f(x,y) - \hat{f}(x,y)]^2$$

Since the square of the quantity is taken, it is always positive. M and N are dimensions of the image. Image quality is also measured as the signal-to-noise ratio (SNR), which can be stated as follows:

$$\text{SNR} = 20 \log_{10} \left(\frac{\text{Signal amplitude}}{\text{Noise amplitude}} \right) \text{dB}$$

To calculate the SNR, normally a good-quality reference image is required. If such an image is not available, then the SNR of an image can be estimated from the given image itself by locating a uniform area that is assumed to be noise free. Now the SNR can be described as follows:

$$\text{SNR} = 20 \log_{10} \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [f(x,y)]^2}{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left[f(x,y) - \hat{f}(x,y) \right]^2} \text{dB}$$

Another useful metric is the peak SNR (PSNR), which can be described for an 8-bit image as follows:

$$\text{PSNR} = 20 \log_{10} \frac{255^2 MN}{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left[f(x,y) - \hat{f}(x,y) \right]^2} \text{dB}$$

The PSNR is measured with respect to peak signal power and the SNR with respect to actual signal power. The unit for both measures is decibels. For a good-quality image, the SNR is very high. However, it is quite possible that an image may have a high SNR but may still not show the information clearly. In such situations, subjective metrics are preferred.

Example 5.1 For reference images

$$f(x,y)=\begin{pmatrix}3 & 2 & 1\\ 1 & 2 & 1\\ 3 & 2 & 2\end{pmatrix} \text{ and } \hat{f}(x,y)=\begin{pmatrix}3 & 1 & 1\\ 1 & 1 & 2\\ 1 & 1 & 1\end{pmatrix},$$

compute the MSE, SNR, and PSNR for an 8-bit image.

Solution

$$\text{MSE}=\frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left[f(x,y)-\hat{f}(x,y)\right]^2$$

$$\text{MSE}=\frac{1}{3^2}\left[(3-3)^2+(2-1)^2+(1-1)^2+(1-1)^2+(2-1)^2+(1-2)^2+(3-1)^2+(2-1)^2+(2-1)^2\right]$$

$$=\frac{1}{9}[9]=1$$

$$\text{SNR}=20\log_{10}\left[\frac{\left(3^2+2^2+1^2+1^2+2^2+1^2+3^2+2^2+2^2\right)}{9}\right]$$

$$=20\log_{10}\left[\frac{37}{9}\right]\cong 12.8$$

$$\text{PSNR}=20\log_{10}\left[\frac{255^2\times 9}{9}\right]\cong 115.35$$

Another useful measure is the contrast-to-noise ratio (CNR), which is denoted by the following formula:

$$\text{CNR}=20\log_{10}\left(\frac{f_{\text{object}}-f_{\text{background}}}{\sigma_f}\right)$$

Here, f_{object} and $f_{\text{background}}$ are the average pixel values of the object and background, respectively; σ_f is the standard deviation of the pixel values. CNR can also be defined as the difference between the SNR of the object and that of the background.

On the other hand, *subjective fidelity criteria* are not based on any metrics. These are, as the name suggests, subjective and depend on the perception of the human observers and their visual systems. This approach is justified on the ground that human observers are the ultimate users. Many standards are available for assessing subjective quality. In general, the quality of an image or a video can be rated based on a score of 0–100. For example,

human observers can be asked to rate the quality, where rating can be defined as follows: 1–19 (bad), 20–39 (poor), 40–59 (fair), 60–79 (good), and 80–100 (excellent). This kind of approach is more suitable for videos. In single-stimulus categorical testing, a random test image can be shown on the screen and users can be asked to rate the image, based on the suggested scale, from 'bad' to 'excellent'. In double-stimulus categorical testing, both reference image and test images are shown for a fixed time. In pair-wise similarity judgement test, two images are shown and users are asked to choose the better image and determine the degree of difference between the images.

Rating depends on different factors such as viewing environment, spatial resolution, optical resolution, light conditions, and colour. The problem with subjective measure is that scores would vary from person to person and, for the same person, at different points of time. To make it more scientific, a group index that indicates the quality of the image can be developed, involving multiple human observers. Measures like difference mean opinion score (DMOS), which finds the difference between the raw quality score of reference and test images, or z-scores can be taken. DMOS is computed as follows:

$$d_{i,j} = r_{i,\text{reference}}(j) - r_i, j$$

Here $r_{i,j}$ is the raw score for the subject i for the image j; $r_{i,\text{reference}}(j)$ is the raw score given by the same subject i for the reference image of the test image j. Then, the z-score can be computed as follows:

$$z_{i,j} = \frac{d_{i,j} - \bar{d}_i}{\sigma_i}$$

where $d_{i,j}$ is the DMOS, d_i is the mean DMOS, and σ_i is the standard deviation for all the images rated by the ith subject.

5.2 IMAGE ENHANCEMENT OPERATIONS

'Transform' is a mapping function that maps a set of input values to its equivalent output values. For example, a square function takes a set of numbers and performs the task of squaring to produce the equivalent set of output values. In image processing, these transforms map a set of input pixel values based on certain manipulations to a set of output pixel values. These transforms are used to manipulate contrast, and hence these operations are often referred to as contrast enhancement, contrast adjustment, or contrast stretching.

Such a transformation that is applied to an image $f(x, y)$ to produce an enhanced image is given mathematically as follows:

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

This is also known as grey-level transformation or spatial transform. Here, $f(x, y)$ is the input image, T is the operator on $f(x, y)$, and $g(x, y)$ is the processed image. As the coordinates are not important for the grey-level transformation, the preceding equation can also be written by ignoring the coordinates, as follows:

$$s = T[r]$$

Here, r is the grey level of the original image and s the resulting grey level after the application of operator T. Thus, T is a grey-level transformation (or mapping) function.

Since the operation involves only one point, it is called a *point transform* or *point operation*. A point operation is also called a *grey-level transformation* or *spatial transform*. The transformation may involve some neighbourhood of (x, y). In this case, T is called a *neighbourhood operation*, and if the transformation involves all the pixels of the image, it can be categorized as a *global operation*.

These operations can be carried out in the spatial or frequency domain. Let us discuss point transformations in the spatial domain now.

5.3 IMAGE ENHANCEMENT IN SPATIAL DOMAIN

In the spatial domain, point transforms or point grey-level scaling transformations depend only on the pixels of the input image to create corresponding pixels in the output image. Point operations are carried out by various scaling functions, which can be categorized as shown in Fig. 5.8.

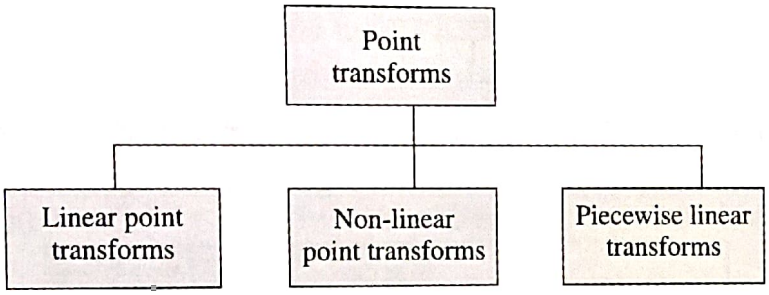


Fig. 5.8 Types of point transforms

Linear functions are of the following form: $a \times f(x, y) + b$, where a and b are constants that manipulate contrast and brightness. Examples of linear functions are identity and inverse. The functions that are not of this form are called non-linear functions. Examples of non-linear functions are square, square root, logarithm, and exponential functions. A piecewise linear function is a set of linear functions that are applied to distinct grey values of the given image.

Scaling functions can be applied directly to the image or the lookup table (LUT). The role of the linear function is to map every grey level value of the original image to a new grey level value in the resultant image. These operations may also involve the LUT. The LUT is a table where results of the mappings are stored. The function refers to the table, takes the result of the mapping from the table, and produces the output pixel. The advantage of this approach is that the original image values are not affected or disturbed directly.

Let us first discuss linear scaling functions.

5.3.1 Linear Point Transformations

Inversion is one of the important linear point transformations, which performs a digital negative operation. In a binary image, inverse transformation reverses the image by changing

a black pixel to a white one and vice versa. This is similar to the NOT operator. However, 8-bit grey scale images can have 256 (2^8) possible different shades of grey, ranging from black to white. The inversion operation is mathematically expressed as follows:

$$g(x, y) = L - 1 - f(x, y)$$

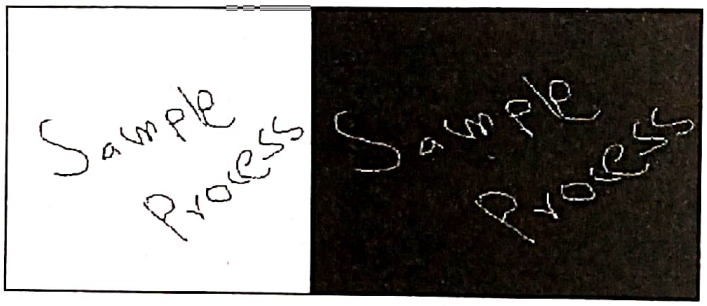
where L is the number of grey levels in the image. Consider the following 3 × 3 image:

$$f(x, y) = \begin{array}{|c|c|c|}\hline 1 & 2 & 3 \\ \hline 0 & 1 & 1 \\ \hline 2 & 2 & 3 \\ \hline \end{array}$$

This image has four grey levels (0, 1, 2, 3). Hence, the inversion transformation can be performed to yield

$$g(x, y) = \begin{array}{|c|c|c|}\hline 2 & 1 & 0 \\ \hline 3 & 2 & 2 \\ \hline 1 & 1 & 0 \\ \hline \end{array}$$

Observe that pixel '0' in the original image is transformed to level '3' and vice versa. A sample image and its inverse are shown in Figs 5.9(a) and (b), respectively.



(a) (b)

Fig. 5.9 Effect of inversion (a) Original image (b) Image negative

Example 5.2 Obtain the digital negative of the following 3 × 3 grey scale image:

122	150	200
225	225	225
250	250	240

Solution Since this is a grey-scale image, the number of grey levels L is $2^8 = 256$.

Therefore,

$$g(x, y) = L - 1 - f(x, y)$$

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

Therefore, the resulting image is given as follows:

133	105	55
30	30	30
5	5	15

Let us now discuss some useful non-linear operators.

5.3.2 Non-linear Point Transformations

For a non-linear operator, the relationship between input and output variables is not linear. Some of the popular non-linear functions frequently used in image processing are discussed in the following subsections.

5.3.2.1 Square Function

This function is used to enhance the contrast of the given image. However, when the pixel values cross the limit accepted by the data type of the image (e.g., integer data type supports 0–255), this operation leads to saturation. Saturation is a condition where all the pixels have high values. Figure 5.10(a) shows an example of how the square of an image causes most of the pixels to cross the limit and hence are represented with the maximum pixel values. The result [Fig. 5.10(b)] is a whitened or saturated image.

(a) (b)

Fig. 5.10 Square of image (a) Original image (b) Resultant image of square function

5.3.2.2 Square Root

The purpose of this function is to expand the grey scale range to the dark areas of the image. In addition, this function reduces the dynamic range of the light areas of the image. The function can be written as follows:

$$g(x,y) = \sqrt{255 \times f(x,y)}$$

Here, $f(x, y)$ is the pixel value of the input image. This function is normally used to display the Fourier spectrum of images. The original image and its square root equivalent image are shown in Figs 5.11(a) and (b).

(a) (b)

Fig. 5.11 Square root of an image (a) Original image
(b) Resultant image of square root operation with edge highlighted

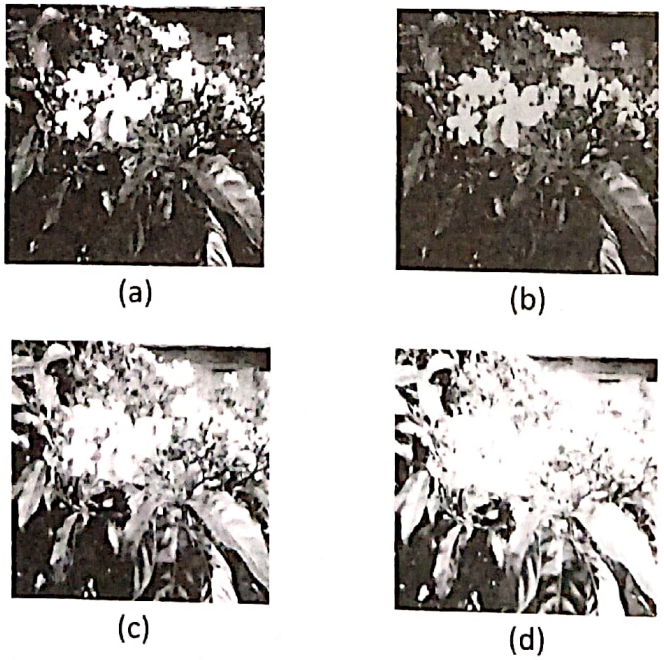
5.3.2.3 Logarithmic Function

The purpose of this function is to compress the dynamic range of images. The logarithmic function is given as follows:

$$g(x, y) = a \times \log[f(x, y)]$$

Here, log(·) is logarithm and $g(x, y)$ is the resultant image of the input image $f(x, y)$.

The advantage of this function is that it increases the dynamic range of dark regions and reduces that of lighter regions. This function is mostly used to view the Fourier-transformed images, as their dynamic ranges are very high. Some of the applications of logarithmic operation are shown in Figs 5.12(a)–(d).



(a) (b)

(c) (d)

Fig. 5.12 Logarithmic operation of image (a) Original image (b) Logarithmic image (a = 1)
(c) Logarithmic image (a = 2) (d) Logarithmic image (a = 3)

As logarithm of zero is not defined, the transformation can be changed as follows:

$$g(x,y) = a \times \log(1 + (e^{\alpha} - 1) f(x,y))$$

Here, $f(x,y)$ is the pixel value of the input image and a is a constant whose value is given as follows:

$$\frac{255}{\log(1 + \max(f(x,y))}$$

The variable a is used to scale the range of the input image and the value of α is to control the input range of the image.

5.3.2.4 Exponential Function

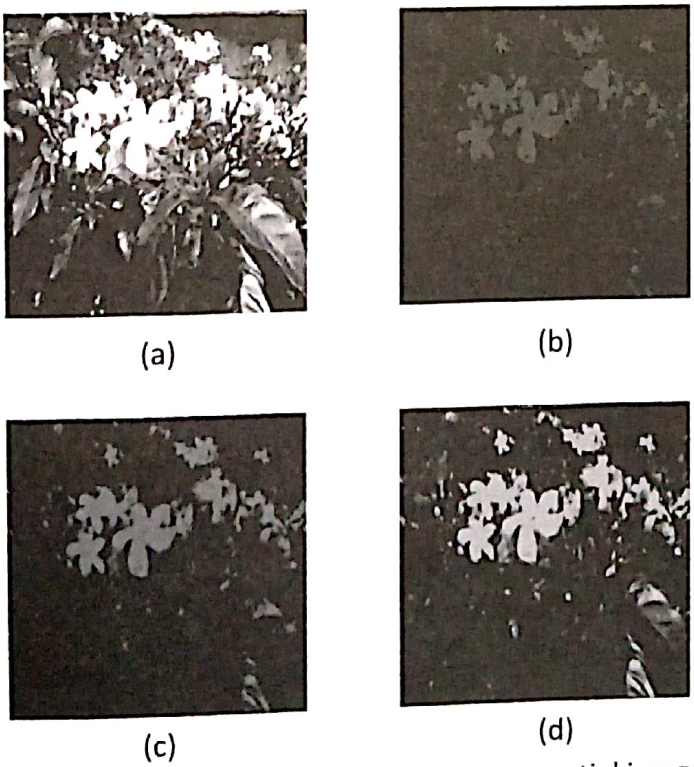
This is the reverse of the logarithmic function and is given as follows:

$$g(x,y) = e^{f(x,y)}$$

The modified exponential function is given as follows:

$$g(x,y) = c \times \left[(1+\alpha)^{f(x,y)} - 1\right] \text{ where } \alpha \text{ is a constant}$$

Here, c is the scaling factor. It can be observed that if $f(x, y) = 0$, then $g(x, y) = 0$, thus showing that one is subtracted as an offset. This function is used to enhance the details in high-value regions of the image and decrease the dynamic range in low-value regions. Some of the applications of the exponential operation are shown in Figs 5.13(a)–(d).



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

Fig. 5.13 Exponential of image (a) Original image (b) Exponential image ($c = 4$ and $\alpha = 0.3$) (c) Exponential image ($c = 4$ and $\alpha = 0.4$) (d) Exponential image ($c = 4$ and $\alpha = 0.6$)

5.3.2.5 Power Function

Another non-linear operator is power transformation, which involves functions of the following form:

$$g = c \times f(x,y)^{\gamma}$$

Here, c and γ are positive constants. If the value of γ is greater than 1, then the contrast of high-value portions is increased at the cost of low-value regions. If the value of γ is less than 1, then the contrast of the high-value portions of the image is reduced at the cost of low-value regions. On the other hand, if the value of γ is equal to 1, then the operation performs range scaling. Thus, depending on the value of γ, this transform is used to stretch or contract the grey-scale values. Some of the applications of power function operation are shown in Figs 5.14(a)–(c).

(a) (b) (c)

Fig. 5.14 Power function (a) Original image (b) Resultant image of power transform (c = 2, γ= 0.3) (c) Resultant image of power transform (c = 2, γ= 0.5)

Example 5.3 Given image

$$f(x,y) = \begin{array}{|c|c|c|c|} \hline 1 & 2 & 3 & 4 \\ \hline 5 & 5 & 6 & 6 \\ \hline 6 & 7 & 6 & 6 \\ \hline 6 & 7 & 2 & 3 \\ \hline \end{array}$$

Assuming that grey level is 0–7, that is, 8, apply the following transformations: inversion, square root, square function, logarithm function with a = 0.5, and power function with c = 1, r = 1.2.

Solution

(a) Inversion/Digital negative operation:

$$g(x,y) = L - 1 - f(x,y)$$

Here, L is the number of grey levels in the image, which is 8.

∴ The inverse image would be as follows:

$$g(x, y) = \begin{array}{|c|c|c|c|}\hline 6 & 5 & 4 & 3 \\ \hline 2 & 2 & 1 & 1 \\ \hline 1 & 0 & 1 & 1 \\ \hline 1 & 0 & 5 & 4 \\ \hline \end{array}$$

(b) Square root operation is $g = \sqrt{7f(x, y)}$. Thus,

$$g(x, y) = \begin{array}{|c|c|c|c|}\hline 2.64 & 3.74 & 4.58 & 5.29 \\ \hline 5.9 & 5.9 & 6.4 & 6.4 \\ \hline 6.4 & 7 & 6.4 & 6.4 \\ \hline 6.4 & 7 & 3.74 & 4.58 \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline 3 & 4 & 5 & 5 \\ \hline 6 & 6 & 6 & 6 \\ \hline 6 & 7 & 6 & 6 \\ \hline 6 & 7 & 4 & 5 \\ \hline \end{array}$$

(c) Square function

$$g(x, y) = f(x, y)^2$$

$$g(x, y) = \begin{array}{|c|c|c|c|}\hline 1 & 4 & 9 & 16 \\ \hline 25 & 25 & 36 & 36 \\ \hline 36 & 49 & 36 & 36 \\ \hline 36 & 49 & 4 & 9 \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline 1 & 4 & 7 & 7 \\ \hline 7 & 7 & 7 & 7 \\ \hline 7 & 7 & 7 & 7 \\ \hline 7 & 7 & 4 & 7 \\ \hline \end{array}$$

As maximum grey level is 7, the values that exceed 7 are rounded off to 7.

(d) Logarithm

$$g(x, y) = a \times \log (f(x, y)) + 1$$

Here, $a = 0.5$. Hence,

$$g(x, y) = \begin{array}{|c|c|c|c|}\hline 1 & 1.15 & 1.23 & 1.3 \\ \hline 1.35 & 1.35 & 1.34 & 1.39 \\ \hline 3.9 & 1.42 & 1.39 & 1.39 \\ \hline 1.39 & 1.42 & 1.15 & 1.23 \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 \\ \hline \end{array}$$

(e) Power function (exponential function)

$$g(x, y) = c \times f^r(x, y)$$

Here, c and r are positive constants whose values are used to stretch or contract the grey scale values. As $c = 1$, $r = 1.2$,

$$g(x, y) = \begin{array}{|c|c|c|c|}\hline 1 & 2.29 & 3.73 & 5.27 \\ \hline 6.89 & 6.89 & 8.58 & 8.58 \\ \hline 8.58 & 10.3 & 8.58 & 8.58 \\ \hline 8.58 & 10.3 & 2.89 & 3.73 \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline 1 & 2 & 4 & 5 \\ \hline 7 & 7 & 9 & 9 \\ \hline 9 & 10 & 9 & 9 \\ \hline 9 & 10 & 2 & 4 \\ \hline \end{array}$$

As grey levels range from 0–7, the resulting image is given as follows:

$$g(x, y) = \begin{pmatrix} 1 & 2 & 4 & 5 \\ 7 & 7 & 7 & 7 \\ 7 & 7 & 7 & 7 \\ 7 & 7 & 2 & 4 \end{pmatrix}$$

5.3.2.6 Gamma Correction

One of the popular applications of using the exponential function is gamma correction or gamma adjustment. The term gamma is used to describe the non-linearity of a computer display monitor. By non-linearity, we mean that the input voltage (V) and output light intensity (I) are related by a power law as follows:

$$I \propto V^{\gamma}$$

or

$$I = kV^{\gamma}$$

where k is a constant and γ is the adjustment made so that the perceived display is linearly related to the input intensity. The variable γ controls the relationship between I and V. When $\gamma = 0$, there is no change in the output and when $\gamma = 1$ or 1.2, the output is roughly linear with the input. Some of the applications of gamma correction are shown in Figs 5.15(a)–(e).

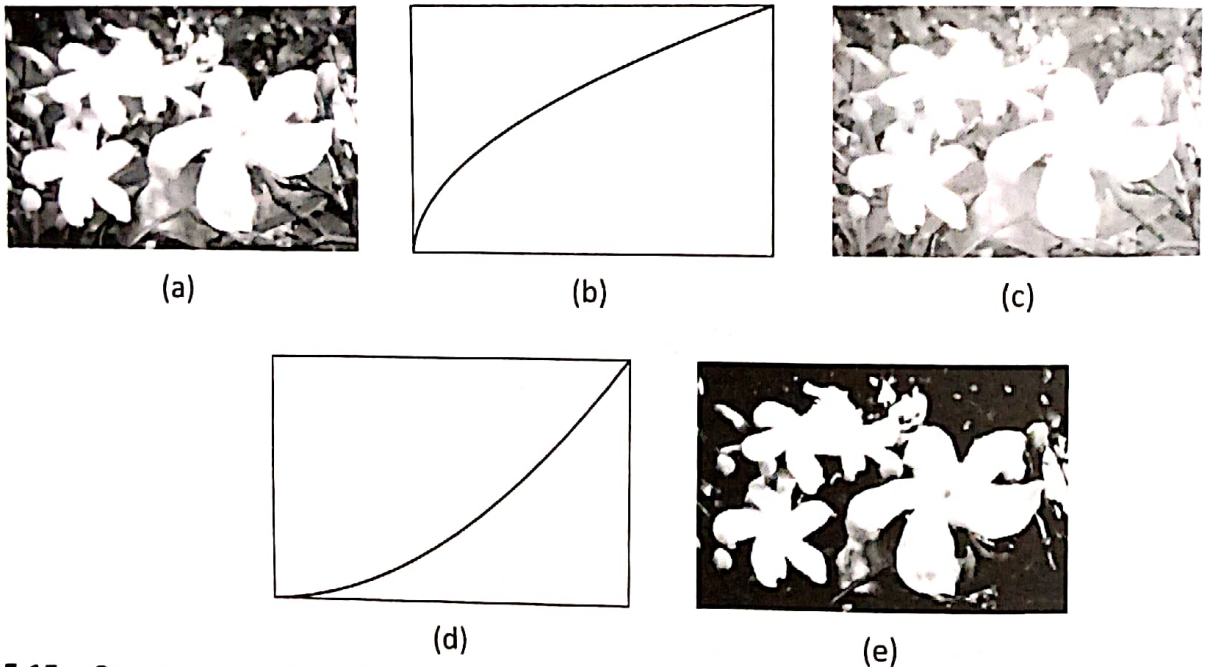


Fig. 5.15 Gamma correction (a) Original image (b) Plot for gamma factor 0.5 (c) Resultant image for gamma factor 0.5 (d) Plot for gamma factor 2.0 (e) Resultant image for gamma factor 2.0

5.3.3 Piecewise Linear Functions

All the functions that have been discussed so far are primitive in nature. Since manipulation of contrast is a difficult subjective process, piecewise linear functions are used to manipulate the contrast of images. Linear functions attempt to improve image quality by manipulating the range of intensity values. Often they stretch the contrast to the fullest possible values to achieve the best visualization of an image. The main advantage of piecewise functions is that they can use simple functions to represent complex functions for intensity transformation. The disadvantage is that they require many input parameters from the user.

5.3.3.1 Contrast Stretching and Its Variants

Contrast stretching is a linear mapping function used to manipulate the contrast of an image. In other words, it maps the values of the input image to the values of the output image. The process can be made efficient by constructing an LUT. The process takes the input value, refers to the LUT, and produces the output value.

Let us assume that the input image has a dynamic range from A to B. Let the grey levels of the input image be r. Very often, the image may not use the full permissible range. Let us assume that the values of A and B are given as 120 and 220, respectively. Therefore, to utilize the full range, it may be necessary to map the pixels to a new range C–D, say 0–255. This kind of mapping process is called contrast stretching. The following is a simple linear piecewise function:

$$s(x,y) = \begin{cases} l \times r \\ m \times (r - r_1) + v \\ n \times (r - r_2) + w \end{cases}$$

The contrast stretching function is shown in Fig. 5.16. Here, l, m, and n are slopes of the function. Contrast stretching can be achieved by controlling the slope and threshold value. When the slope is less than 1, the dynamic range is reduced. The reduction in dynamic ranges results in a condition where the dark regions become darker. When the slope is greater than 1, the brightness of bright regions is further increased. Thus, by controlling the parameters of the function, the contrast of an image can be controlled.

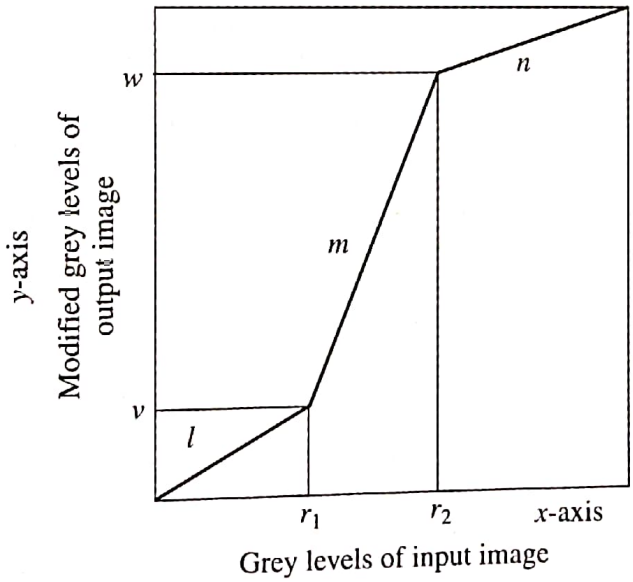


Fig. 5.16 Contrast stretching function

Example 5.4 What will be the dynamic range of an image if the slopes of the image are given as $l = 0.2$, $m = 0.5$, and $n = 0.1$?

Solution Let us assume that the grey levels (r) of the input image range from 0 to 7. The contrast stretching function is shown in Fig. 5.16. Here, the x-axis shows the grey levels of the input image,

and the y-axis shows the modified grey levels of the output image. Let l, m, and n be the slopes of the contrast function. In the contrast stretching function shown in Fig. 5.16, the slopes are given as $l = 0.2$, $m = 0.5$, and $n = 0.1$. Let us assume that $r_1 = 2$ and $r_2 = 5$.

Since the slope $l = 0.2$, the output for grey values $r = 0$, 1, and 2 (i.e., up to r_1) is given as follows:

$$r = 0;\ 0.2 \times 0 = 0$$
$$r = 1;\ 0.2 \times 1 = 0.2$$
$$r = 2;\ 0.2 \times 2 = 0.4$$

For grey levels ($r = 3$, 4, 5, i.e., values between r_1 and r_2), the slope m is given as 0.5 and the transformation is given as $m \times (r - r_1) + v$. The value v is obtained as the last output value of r_1. Therefore, the output values are given as follows:

$$r = 3;\ 0.5 \times (3 - 2) + 0.4 = 0.9$$
$$r = 4;\ 0.5 \times (4 - 2) + 0.4 = 1.4$$
$$r = 5;\ 0.5 \times (5 - 2) + 0.4 = 1.9$$

For grey levels ($r = 6$, 7, i.e., values greater than r_2), the slope n is given as 0.1 and the transformation is given as $n \times (r - r_2) + w$. The value w is obtained as the last output value of r_2.

$$r = 6;\ 0.1 \times (6 - 5) + 1.9 = 2$$
$$r = 7;\ 0.1 \times (7 - 5) + 1.9 = 2.1$$

This shows that for the input range $r = 0$–7, the output range is $s = 0$–2.1. Since the dynamic range is reduced, the image will become darker. This is because the slope values are chosen as fractions that are less than 1.

Many variants of contrast stretching are available. Some of these variants such as range normalization, clipping, binarizing (also known as thresholding), and posterizing (also known as multiple thresholding) form a part of the following discussion.

Range normalization Range normalization is a special form of contrast stretching, which is used to enhance image quality by altering the range of grey levels present. Let us assume that the grey levels of an image range from a to b. If the contrast range is required to be changed as c–d, then the change can be accomplished by the following steps:

1. Subtract a from each grey level so that the range now becomes $0 - (b - a)$.
2. Multiply the result by $\dfrac{(d-c)}{(b-a)}$. Now the range becomes $0 - (d - c)$.
3. Add c to the result. This results in the range c–d.

This transformation can now be described as follows:

$$g(x, y) = \left[\frac{(d-c)}{(b-a)}\right] \times [f(x, y) - a] + c$$

Thus, the input image of range (a–b) can be transformed to an output image with range (c–d) using this transformation. Some of the applications of contrast stretching are shown in Figs 5.17(a) and (b).

(a) (b)

Fig. 5.17 Range normalization (a) Original image (b) Contrast stretched image

Example 5.5 A grey-level image is given, whose range is 10–60. It is necessary to transform this image to another image B, whose range should be 120–180. What should be the grey-level transformation? Let the image be $\begin{pmatrix} 10 & 15 \\ 20 & 50 \end{pmatrix}$

Solution Here, $a = 10$, $b = 60$, $c = 120$, and $d = 180$. Hence,

$$g(x, y) = \left[\frac{(180-120)}{(60-10)}\right] \times [f(x, y) - 10] \div 120$$

$$= 1.2 \times [f(x, y) - 10] \div 120$$

Therefore, the required grey-level transformation is given as follows:

$$1.2 \times [f(x, y) - 10] \div 120$$

Therefore, applying the grey-level transformation results in the image

$$g(x, y) = \begin{array}{|c|c|} \hline 120 & 126 \\ \hline 132 & 168 \\ \hline \end{array}$$

Clipping and thresholding Low-contrast images can be enhanced by simply cutting off very light and very dark areas. Clipping is a special type of contrast stretching where the image is supposed to be in the range [0 … L]. The grey scaling function can be written as follows:

$$G = \begin{cases} 0 & \text{if } 0 \le f(x,y) < t_0 \\ \dfrac{255}{t_1 - t_0} & \text{if } t_0 \le f(x,y) < t_1 \\ L & \text{if } f(x,y) \ge t_1 \end{cases}$$

Here, the thresholds (t_0 and t_1) can be determined from the histogram of the given image. One special type of clipping is the thresholding process, where the two thresholds are the same. In thresholding, the transformation takes every pixel of the image and compares it with the threshold operation to get the corresponding pixel in the resultant image. The transformation can be written as follows:

$$g(x,y) = \begin{cases} 0 & \text{if } f(x,y) \le T \\ 1 & \text{otherwise} \end{cases}$$

Here, T is the user-given threshold value. For example, for the image shown in Fig. 5.18(a), for $T = 3$, the resultant image would be as given in Fig. 5.18(b).

2	2	1
1	1	2
1	1	2

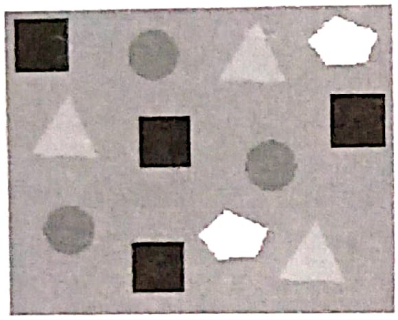
(a)

0	0	0
0	0	0
0	0	0

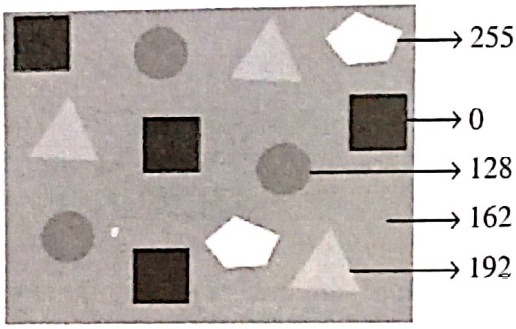
(b)

Fig. 5.18 Illustration of threshold process (a) Original image (b) Threshold image for $T = 3$

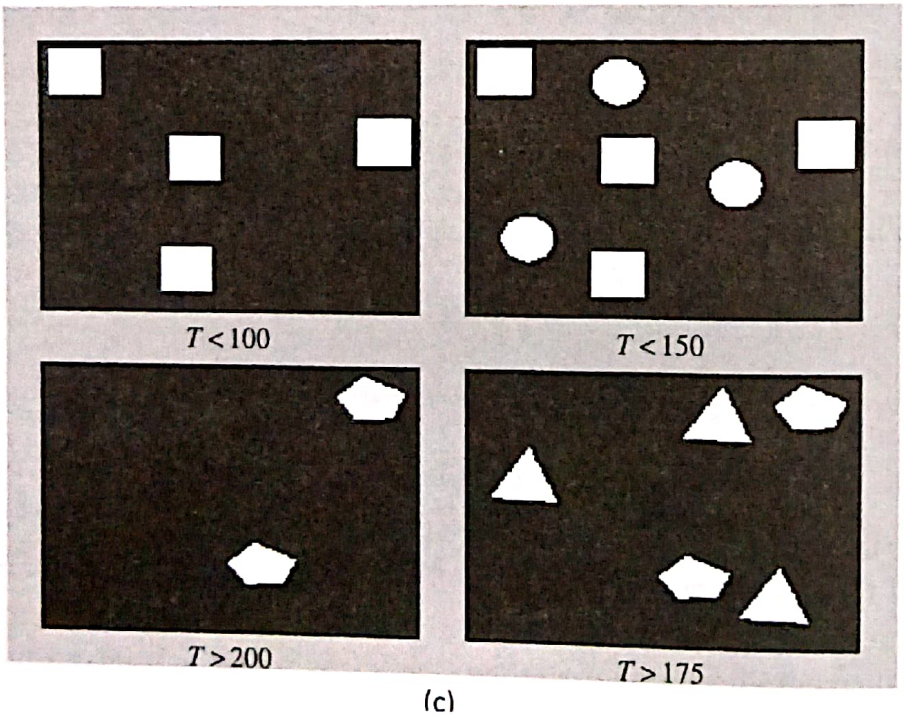
This results in a blank image as the threshold value is too high. An optimal value of the threshold ensures good quality of the resultant image. Sometimes, multiple thresholds may be required to get the resultant image. An example of the thresholding operation is illustrated in Figs 5.19(a)–(c). It can be observed that the selection of the threshold value is a difficult process. Based on the choice of the threshold value, the result also changes.



(a)



(b)



(c)

Fig. 5.19 Thresholding (a) Original image (b) Pixel intensity values (c) Resultant images obtained using different thresholds

Binarization method is a special variant of thresholding where two thresholds are used. If the pixel value falls within these thresholds, the value is set to the maximum level, say 255, in a grey-scale image. All other pixels outside the window are set to zero. This operation is called a windowing operation or binarizing operation.

Posterizing or multiple thresholding This method is an extension of the thresholding. In thresholding, only one threshold is used. However, in this method, multiple thresholds are used to map an image to its resultant image.

$$g = \begin{cases} g_1 & \text{if} \quad 0 \le f_i \le t_1 \\ g_2 & \text{if} \quad t_1 \le f_i \le t_2 \\ \vdots & \qquad \vdots \\ g_n & \text{if} \quad t_{n-1} \le f_i \le 255 \end{cases}$$

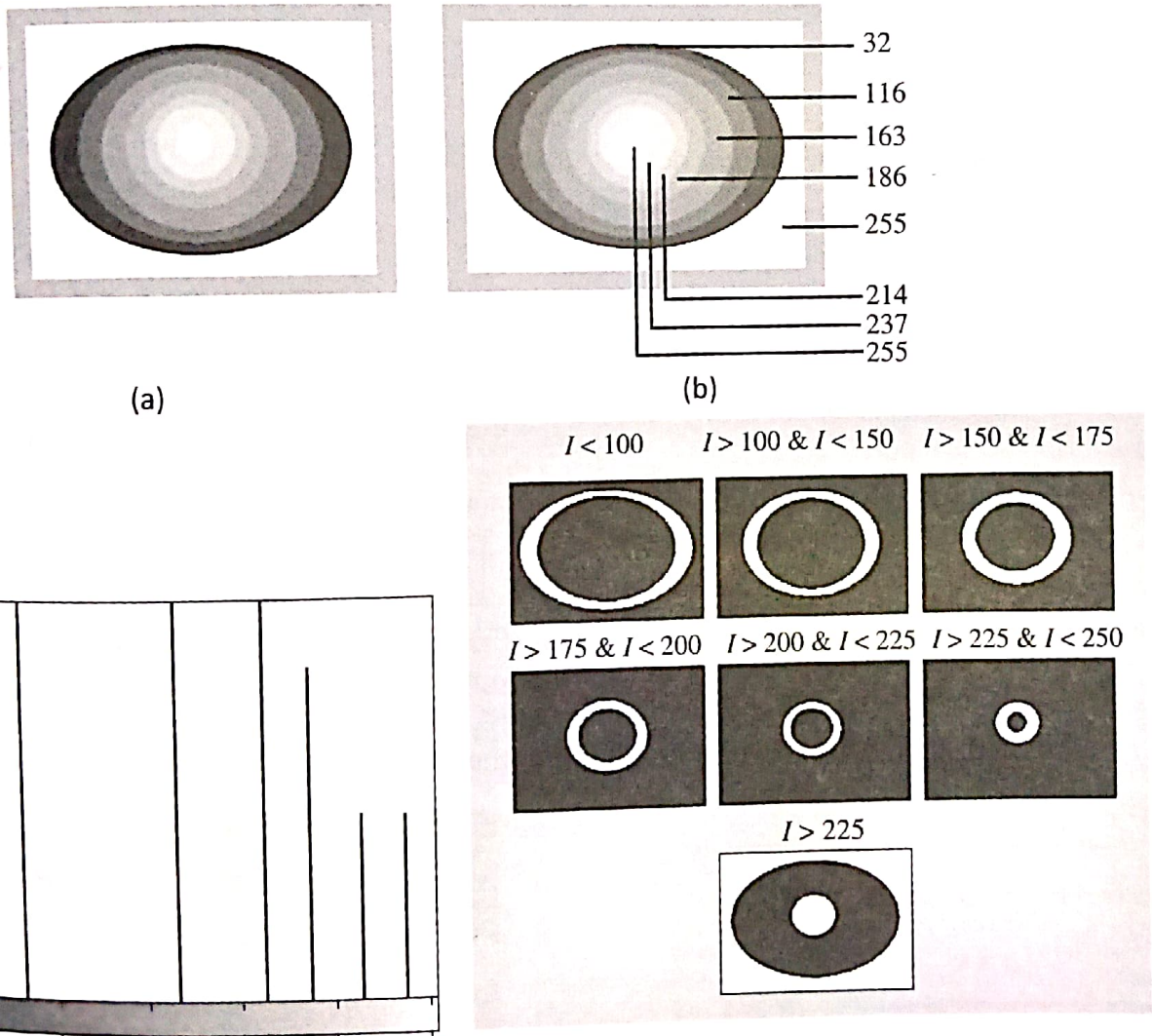


Fig. 5.20 Posterizing (a) Original image (b) Pixel intensity values (c) Histogram of original image (d) Resultant images obtained using different thresholds

Figures 5.20(a)–(d) illustrate this method. Figure 5.20(c) clearly shows that the histogram has many peaks and a large valley between the peaks. The valley indicates the threshold value. Figure 5.20(d) shows the results of the different thresholds.

5.3.3.2 Intensity Slicing

Sometimes, it is necessary to highlight a specific range of intensity levels in an image. This is called the region of interest (ROI). It is often required to enhance only the ROI and suppress the other features present in the image. For example, if the ROI appears bright in a dark background, the brightness of the ROI can be further increased and the darkness of the background can be further decreased, thus increasing the contrast.

Example 5.6 Perform grey-level slicing on the following image:

3	4	5
6	6	7
1	2	2

Solution Grey-level slicing is done as follows:

(a) Without background

The transformation is given as follows:

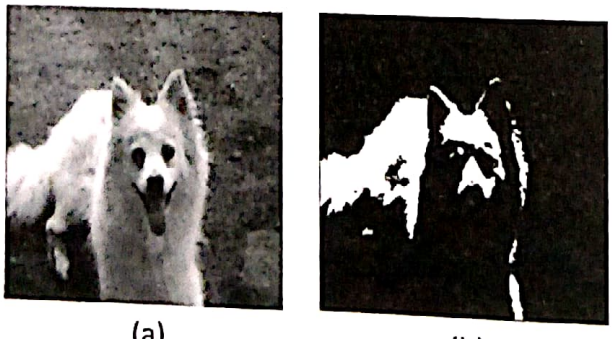
$$S = \begin{cases} L-1 & \text{when } r_1 \le r \le r_2 \\ 0 & \text{otherwise} \end{cases}$$

This results in the following image for $r_1 = 3$ and $r_2 = 6$:

7	7	7
7	7	0
0	0	0

It can be observed that the pixels in the range 3–6 are transformed to the level $L-1$, that is, $8-1=7$, while all other pixels become 0.

Figure 5.21(a) shows the original image, and Fig. 5.21(b) shows the result of the grey-slicing operation. It can be observed that the background information is lost.



(a) (b)

Fig. 5.21 Grey-level slicing without background preservation (a) Original image (b) Resultant image with background not preserved

(b) With background

To preserve the background, the following function is used:

$$S = \begin{cases} L-1 & \text{when } r_1 \le r \le r_2 \\ r & \text{otherwise} \end{cases}$$

This results in the following image:

7	7	7
7	7	7
1	2	2

It can be observed that the pixels in the range 3–6 are transformed to the level $L - 1$, that is, $8 - 1 = 7$, while all other pixels retain their original value. Figure 5.22(a) shows the original image and Fig. 5.22(b) shows the result of the grey-slicing operation. It can be observed that the background information is preserved.

(a) (b)

Fig. 5.22 Intensity slicing with background preservation (a) Original image (b) Resultant image with background preserved

5.3.3.3 *Bit-plane Slicing*

It is possible to transform a grey-level image into a sequence of binary images by taking advantage of its bit pattern storage. For example, a grey-level image is stored as an 8-bit image. Therefore, the image can be transformed into an eight-level image where the zero planes consist of the last bit of each grey level. The first plane consists of the first bit of each grey value. Similarly, the most significant bit plane can be constructed using the most significant bit. The most significant bit plane has the greatest effect in terms of the magnitude of the image. It is roughly equivalent to the threshold of the image at level 127. Depending on the requirement, the bit planes can be retained or ignored.

Example 5.7 Show the bit-plane slicing of the following image:

7	6	5
4	3	2
1	1	0

Solution The following is the binary equivalent of the pixels:

111	110	101
100	011	010
001	001	000

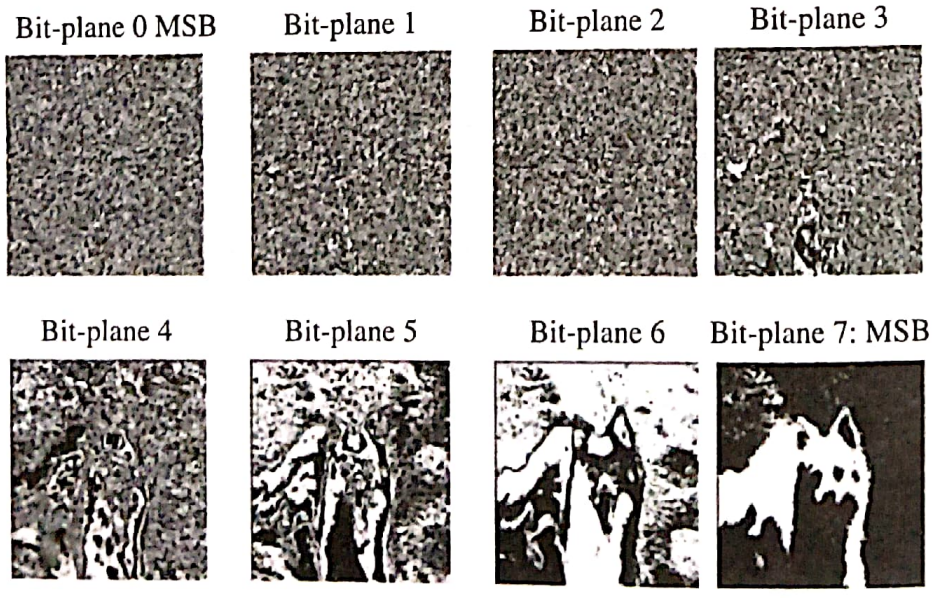
Suppose the least significant bit (LSB) is changed to 0, then the image is reduced to the following form:

110	110	100
100	010	010
000	000	000

The image can be further reduced to the following form:

6	6	4
4	2	2
0	0	0

(a)



(b)

Fig. 5.23 Bit Slicing (a) Original image (b) All eight planes of original image

Suppose the most significant bit (MSB) is changed to 0; this image is reduced to the following form:

011	010	000
000	011	010
001	001	000

This reduces the image to the following format:

3	2	0
0	3	2
1	1	0

A sample image and its extracted planes are shown in Figs 5.23(a) and (b).

5.4 HISTOGRAM-BASED TECHNIQUES

Histogram techniques are very effective and useful in many image processing applications. As discussed earlier, a histogram is an effective tool not only for image quality assessment, but also for manipulating the contrast and brightness of an image. A histogram can be visualized as an intensity distribution or a probability density function. The histogram for a good image will have a flat profile or distribution of pixels. It means that the pixel count is roughly the same for all intensities. The cumulative histogram can also be constructed from the individual histograms by taking the sum of the pixel counts. This can be visualized as a cumulative density function the slope of which is always positive. This can be used as a mapping function for adjusting the contrast, similar to the process in the contrast stretching method. Thus, the quality of an image can be controlled indirectly by normalizing its histogram to a flat profile or by transforming it into a target histogram profile.

5.4.1 Histogram Stretching

The aim of this transformation is to spread the histogram to cover the entire dynamic range, instead of changing the shape of the histogram. This operation is also known as histogram scaling. The mapping function for histogram stretching is given as follows:

$$\frac{S_{max} - S_{min}}{r_{max} - r_{min}}$$

and the transform is given as follows:

$$\frac{S_{max} - S_{min}}{r_{max} - r_{min}}(r - r_{min}) + S_{min}$$

Here, S_{max} and S_{min} are the maximum and minimum of the pixel values in the stretched histogram, respectively, and r_{max} and r_{min} are the maximum and minimum grey-level values in the original image, respectively. This transformation improves the contrast. Figure 5.24(a) shows an original image. It can be observed that the original image is dark; the histogram of the original image is shown in Fig. 5.24(b).

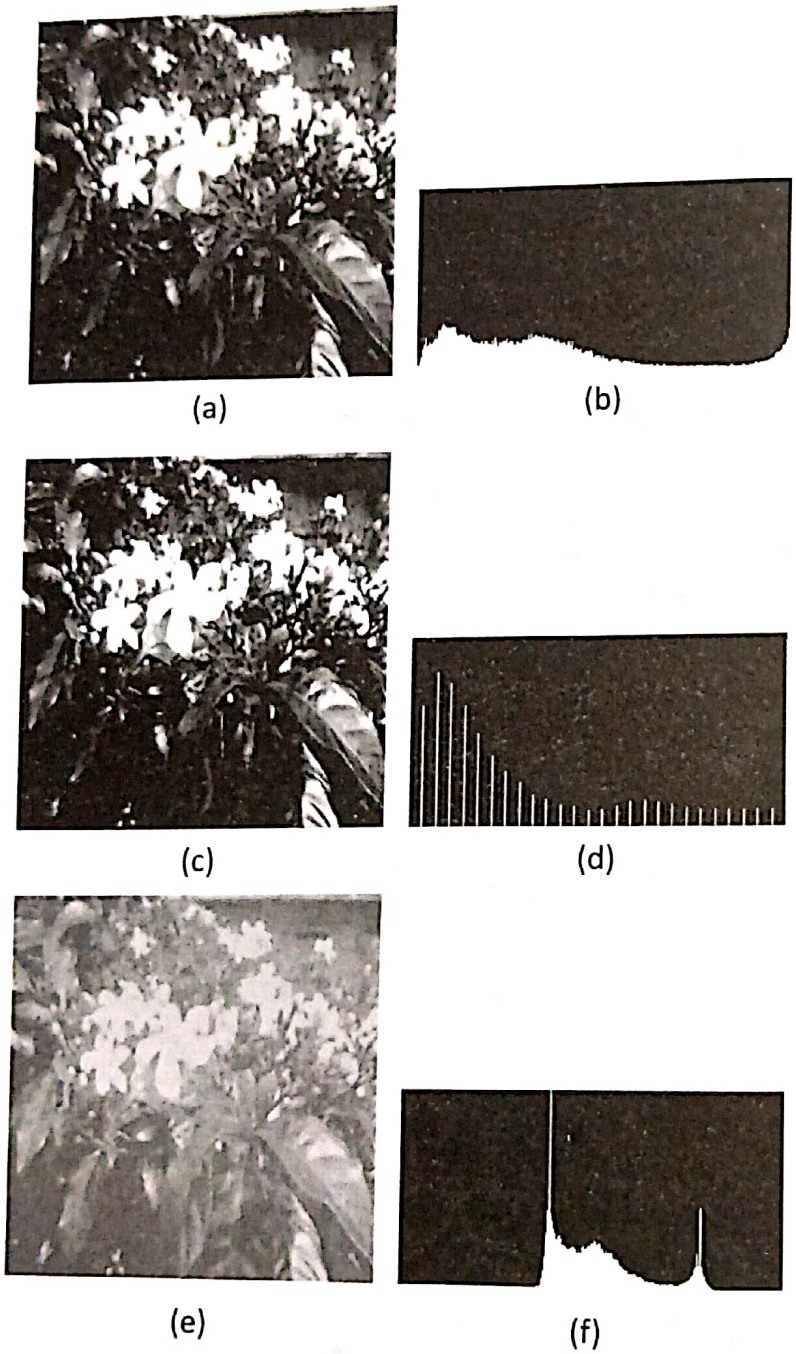


Fig. 5.24 Histogram stretching (a) Original image (b) Histogram of original image (c) Stretched image (d) Histogram of stretched image (e) Shrink image (f) Histogram of shrink image

Figure 5.24(c) shows a stretched image, and the histogram of the stretched image is shown in Fig. 5.24(d). Similarly, Fig. 5.24(e) shows a shrink image, and the histogram of the shrink image is shown in Fig. 5.24(f).

Example 5.8 Perform histogram stretching on the 8 × 8, eight-level grey image, the grey-level distribution of which is shown in Table 5.3.

Table 5.3 Image grey-level distribution (Example 5.8)

Grey levels (r_k)	0	1	2	3	4	5	6	7
Number of pixels (p_k)	0	0	5	20	20	19	0	0

Solution It can be observed that the image ranges from grey level 2 to grey level 5 only. This has to be stretched to cover the entire range 0–7 for enhancing the quality of the image.

Now, $r_{min} = 2$, $r_{max} = 5$, $S_{min} = 0$, and $r_{max} = 7$. Therefore,

$$S = \frac{7-0}{(5-2)}[r-2]+0$$

$$= \frac{7}{3}[r-2]$$

When $r = 2, S = 0$ $\left\{\text{i.e., } \frac{7}{3}[2-2]\right\}$

$r = 3, S = 2.33 \simeq$ (grey level 2)

$r = 4, S = 4.66 \simeq$ (grey level 5)

$r = 5, S = 7$

$r = 6, S = 7$

$r = 7, S = 7$

The resultant mapping will be $S = 0$ corresponding to $r = 2$. Therefore, the pixels that have grey level $r = 2$ in the original image have to be mapped to level $S = 0$ in the resultant image. The final mappings are shown in Table 5.4.

Table 5.4 Final grey-level distribution

Grey levels (S_k)	0	1	2	3	4	5	6	7
Number of pixels (p_k)	5	0	20	0	0	20	0	19

5.4.2 Histogram Sliding

This operation can make an image either darker or lighter, but retains the relationship between grey-level values. It can be represented as follows:

$$S = \text{slide}\,(r) = r + \text{offset}$$

Here, an offset is the extent used to slide the histogram. A positive value increases brightness, and a negative value creates a darker image. Figure 5.25(a) shows an original image. It can be observed that the original image is dark. The result of histogram sliding by an offset of 30 is shown in Fig. 5.25(b). It can be observed that the resulting image is brighter than the original image and reveals more details.

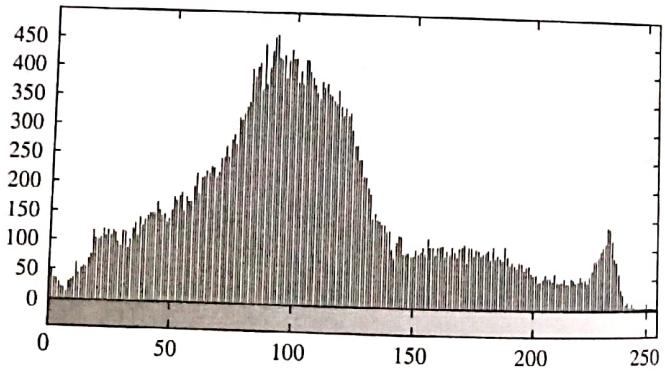
(a) (b)

Fig. 5.25 Histogram sliding (a) Original image (b) Resultant image (histogram sliding by 30 units)

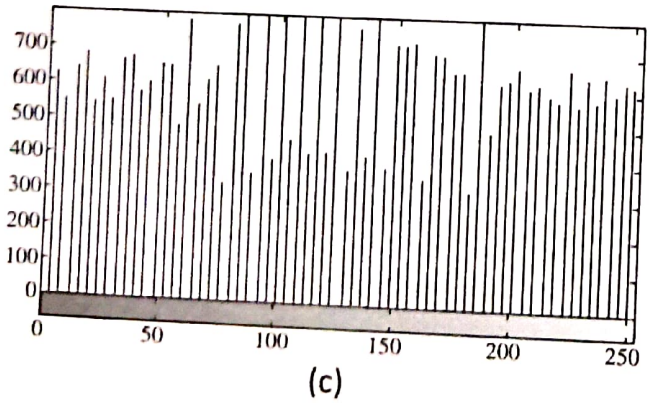
5.4.3 Histogram Equalization

This technique is similar to histogram stretching. It tries to flatten the histogram to create a better quality image. It treats an image as a probability distribution. It first finds the cumulative distributive function, after which it normalizes the values and performs respective mapping. The histogram equalization process is summarized in the following steps:

(a)



(b)



(c)

(d)

Fig. 5.26 Histogram equalization (a) Original image (b) Histogram of original image (c) Equalized histogram (d) Resultant image

1. Form the cumulative histogram.
2. Normalize the value by dividing it by the total number of pixels.
3. Multiply the value by the maximum grey-level value and round off the value.
4. Map the original value to the result of step 3 by a one-to-one correspondence.

This process will equalize the histogram. Histogram equalization may not always provide the desired effect because its aim is to distribute the pixels evenly. Figure 5.26(a) shows an original image. It can be observed that the original image is dark. The histogram of the image is shown in Fig. 5.26(b). The equalized histogram and the resultant image are shown in Figs 5.26(c) and (d), respectively.

Example 5.9 Perform histogram equalization for the 8 × 8, eight-level image described in Table 5.5.

Table 5.5 Pixel distribution of the image

r_k	0	1	2	3	4	5	6	7
p_k	8	10	10	2	12	16	4	2

Solution The equalization process for the given image is shown in Table 5.6.

Table 5.6 Image equalization process (Example 5.9)

r_k	p_k	Cumulative running of pixels	$\frac{\text{Cumulative}}{\text{Total}} \times (L-1)$	Round off to the nearest grey level
0	8	8	8/64 × 7 = 0.875	1
1	10	18	18/64 × 7 = 1.968	2
2	10	28	28/64 × 7 = 3.0625	3
3	2	30	30/64 × 7 = 3.2812	3
4	12	42	42/64 × 7 = 4.5937	5
5	16	58	58/64 × 7 = 6.3437	6
6	4	62	62/64 × 7 = 6.78125	7
7	2	64	64/64 × 7 = 7	7

Table 5.7 shows the final distribution of grey scale pixels where pixels 28 and 30 corresponding to levels 2 and 3, respectively, are mapped to level 3 in the resultant image. Similarly, pixels 62 and 64 are mapped to level 7 in the resultant image.

Table 5.7 Final grey-level distribution

Grey levels (r_k)	0	1	2	3	4	5	6	7
Number of pixels (p_k)	1	2	3	3	5	6	7	7

Example 5.10 Perform histogram equalization on the following 3×3, eight-level image:

$$\begin{pmatrix} 1 & 3 & 5 \\ 4 & 4 & 3 \\ 5 & 2 & 2 \end{pmatrix}$$

Solution The histogram equalization process for the given image is shown in Table 5.8.

Table 5.8 Image equalization process (Example 5.10)

r_k	p_k	Cumulative running of pixels	$\frac{\text{Cumulative}}{\text{Total}} \times (L-1)$	Round off to the nearest grey level
0	0	0	$0/9 \times 7 = 0$	0
1	1	1	$1/9 \times 7 = 0.777$	1
2	2	3	$3/9 \times 7 = 2.333$	2
3	2	5	$5/9 \times 7 = 3.888$	4
4	2	7	$7/9 \times 7 = 5.444$	5
5	2	9	$9/9 \times 7$	7
6	0	9	$9/9 \times 7$	7
7	0	9	$9/9 \times 7$	7

The image after equalization is as follows:

$$\begin{pmatrix} 1 & 4 & 7 \\ 5 & 5 & 4 \\ 7 & 2 & 2 \end{pmatrix}$$

Example 5.11 Apply the histogram equalization technique to the following image:

$$f(x, y) = \begin{array}{|c|c|c|c|} \hline 1 & 2 & 3 & 4 \\ \hline 5 & 5 & 6 & 6 \\ \hline 6 & 7 & 6 & 6 \\ \hline 6 & 7 & 2 & 3 \\ \hline \end{array} .$$

Solution This is explained in Tables 5.9 and 5.10.

Table 5.9 Pixel distribution of image

Grey levels (r_k)	0	1	2	3	4	5	6	7
No. of pixels (p_k)	0	1	2	2	1	2	6	2

Table 5.10 Image equalization process

r_k	p_k	Cumulative	$\frac{\text{Cumulative}}{\text{Total}} \times (L-1)$	Round-off value
0	0	0	$0/16 \times 7 = 0$	0

(Contd)

r_k	p_k	Cumulative	$\frac{\text{Cumulative}}{\text{Total}\times(L-1)}$	Round-off value
1	1	1	$1/16\times 7 = 0.4375$	0
2	2	3	$3/16\times 7 = 1.3125$	1
3	2	5	$5/16\times 7 = 2.1875$	2
4	1	6	$6/16\times 7 = 2.625$	3
5	2	8	$8/16\times 7 = 3.5$	4
6	6	14	$14/16\times 7 = 6.125$	6
7	2	16	$16/16\times 7 = 7$	7

The image after histogram equalization is as follows:

0	1	2	3
4	4	6	6
6	7	6	6
6	7	1	2

5.4.4 Histogram Specification

A user has no control over the histogram normalization process. However, histogram equalization allows exercising control over the process through target histogram specification. The algorithm for histogram specification is as follows:

1. Find the mapping table of histogram equalization.
2. Specify the desired histogram. Equalize the desired histogram.
3. Perform the mapping process so that the values of Step 1 can be mapped to the results of Step 2.

Example 5.12 Perform histogram specification on the 8 × 8, eight-level image described in Table 5.11.

Table 5.11 Pixel distribution of the image

p_k	8	10	10	2	12	16	4	2
r_k	0	1	2	3	4	5	6	7

The target histogram is as shown in Table 5.12.

Table 5.12 Pixel distribution of the image

r_k	0	1	2	3	4	5	6	7
p_k	0	0	0	0	20	20	16	8

Solution The equalization process for this image has already been carried out and the result is shown in Table 5.7.

The next step is to equalize the target histogram. The target histogram is equalized as shown in Table 5.13.

Table 5.13 Equalization of the target histogram (Example 5.12)

r_k	p_k	Cumulative running of pixels	Cumulative / Total $\times(L-1)$	Round off to the nearest grey level
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	20	20	$20/64 \times 7 \simeq 2.1875$	2
5	20	40	$40/64 \times 7 = 4.375$	4
6	16	56	$56/64 \times 7 = 6.125$	6
7	8	64	$64/64 \times 7 = 7$	7

The mapping is shown in Table 5.14.

Table 5.14 Mapping obtained from equalization process

r_k	Closest level
0	0
1	0
2	0
3	0
4	2
5	4
6	6
7	7

Now map the values from the source to the target. The final mapping between the source and the target histograms is shown in Table 5.15.

Table 5.15 Final mapping process

Rows (grey levels)	H (mappings of the equalization)	S (mappings of the equalization of the target)	Map
0	1	0	4
1	2	0	4
2	3	0	5
3	3	0	5
4	5	2	6
5	6	4	6
6	7	6	7
7	7	7	7

Now, the resultant mapping can be found by taking the equalized value H and searching column S. In this case, take a value, say 1, and search in the specification column. In column S, the closest value is 2, which is present in row 4. Hence, the new mapping is 4. In the case of a clash, the choice should be the largest value, in order to provide maximum contrast. However, by picking the smallest value, a more gradually changing image is obtained.

An application of histogram specification is shown in Figs 5.27(a)–(e). The original image is shown in Fig. 5.27(a), and the histogram of the image is shown in Fig. 5.27(b). The target histogram is shown in Fig. 5.27(c). The resultant obtained after carrying out histogram specification is shown in Fig. 5.27(d). The resultant image obtained after histogram specification is shown in Fig. 5.27(e).

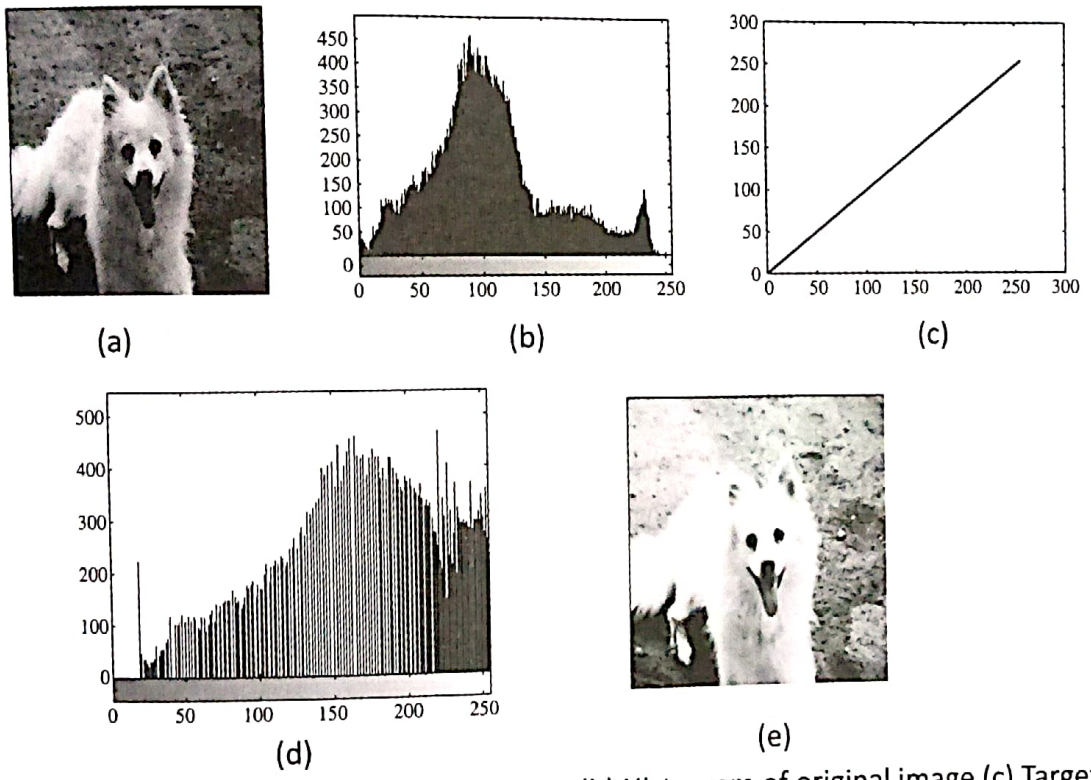


Fig. 5.27 Histogram specification (a) Original image (b) Histogram of original image (c) Target histogram (d) Specified histogram (e) Resultant image after specification

Example 5.13 Apply histogram specification to the image given here. Assume the target given in Table 5.16. Show the resultant final mappings.

$$I_2 = \begin{array}{|c|c|c|c|}\hline 1 & 3 & 4 & 5 \\ \hline 5 & 6 & 6 & 6 \\ \hline 7 & 7 & 7 & 7 \\ \hline 5 & 5 & 5 & 5 \\ \hline \end{array}$$

Solution First, perform histogram equalization for this image. Let the target mapping for grey levels 0–7 be 0, 0, 1, 2, 2, 3, 6, 7.

Table 5.16 Histogram equalization process

r_k	p_k	Cumulative	$\frac{\text{Cumulative}}{\text{Total}\times(L-1)}$	Round-off value
0	0	0	$0/16\times 7=0$	0
1	1	1	$1/16\times 7=0.4375$	0
2	0	1	$1/16\times 7=0.4375$	0
3	1	2	$2/16\times 7=0.875$	1
4	1	3	$3/16\times 7=1.3125$	1
5	6	9	$9/16\times 7=3.93$	4
6	3	12	$12/16\times 7=5.25$	5
7	4	16	$16/16\times 7=7$	7

By combining the results of Example 5.12, one can get the final mapping as shown in Table 5.17.

Table 5.17 Final mapping

Grey level	$H(I_1)$	$H(I_2)$	Mapping
0	0	0	1
1	0	0	1
2	0	1	1
3	1	2	2
4	1	2	2
5	4	3	5
6	5	6	6
7	7	7	7

By applying these mappings, one can get the final image as follows:

1	2	5	5
5	5	7	7
7	7	7	7
7	7	2	5

5.4.5 Local and Adaptive Contrast Enhancement

The histogram techniques that have been discussed so far are global techniques as they are applied to the entire image. Sometimes, it may be necessary to enhance the local details of an image. In such situations, local operations are preferred to global operations. As discussed in Section 5.2, a local operation requires the concept of a local neighbourhood, which is specified by a window. The local histogram enhancement algorithm can be stated as follows:

1. Select a neighbourhood of an image using a spatial window.
2. Apply histogram equalization or histogram specification to the selected neighbourhood pixels.
3. Move the window. Hence, the centre of the neighbourhood is moved by a pixel.
4. Update the histogram.
5. Repeat steps 2–4 until the entire image is processed.

This approach is tedious as more computations are required, but the results of this algorithm are accurate.

Adaptive filters can also be used for enhancing images of uneven contrast. Adaptive filter behaviour varies based on local image statistics such as mean and variance. Similar to local enhancement, a window is used to select the neighbourhood. When the dimensions of the window equal those of the image, the algorithm becomes a global algorithm. Generally, it is a trade-off between the dimensions of the window and the effectiveness of the method, as a mask of a smaller size increases the number of computations by a great deal. Adaptive contrast manipulation is carried out using the following formula:

$$\left[\left[k_1 \times \left(\frac{\mu_f}{\sigma_i}\right) \times (f(x,y) - \mu_i)\right] + (k_2 \times \mu_i)\right]$$

Here, μ_f is the mean of the entire image; μ_i and σ_i are, respectively, the standard deviation and local mean specified by the window; and k_1 and k_2 are constants whose values range between 0 and 1. The process involves subtraction of the local mean from the image and weighing the result by the gain factors k_1 and k_2 that can be controlled. Then the window is moved and the processes are repeated till the entire image is processed. The original image is shown in Fig. 5.28(a) and the results of the adaptive contrast enhancement are shown in Figs 5.28(b) and (c).

Fig. 5.28 Adaptive contrast enhancement (a) Original image (b) Enhanced image with exponential distribution (c) Enhanced image with uniform distribution

This algorithm is often used to intensify local variations and boost the contrast of low-contrast regions.

5.5 SPATIAL FILTERING CONCEPTS

Many image enhancement techniques are based on spatial operations that are performed on the local neighbourhood of the input pixels. The objective of the neighbourhood operation in image processing is illustrated in Fig. 5.29.

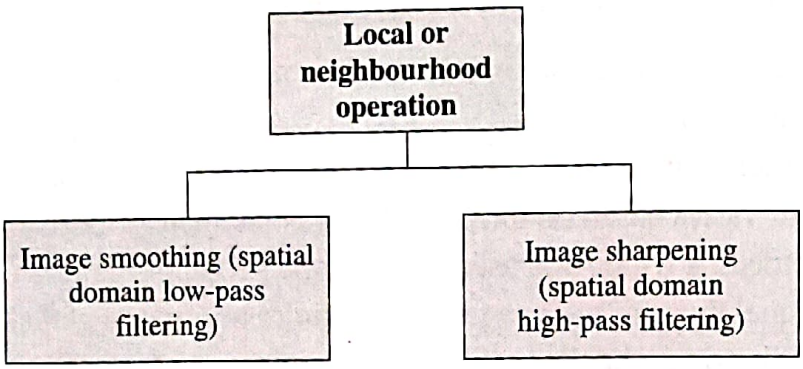


Fig. 5.29 Types of local or neighbourhood operations

Image smoothing and sharpening are illustrated in Fig. 5.30. Figure 5.30(a) shows the original image, and Fig. 5.30(b) shows blurring of the image. Figure 5.30(c) shows the sharpened image of the original image shown in Fig. 5.30(a). It can be seen that blurring leads to smoothing as well as loss of sharp details. On the other hand, image sharpening leads to sharpened edges and enhanced details. These kinds of operations are implemented using neighbourhood operations in the form of a filter.

(a) (b) (c)

Fig. 5.30 Image smoothing and sharpening (a) Original image (b) Blurring of original image (c) Sharpening of original image

Neighbourhood operations are carried out by filters. Filtering is a technique with which certain frequency components can be chosen or rejected. For example, a low-pass filter selects only the low-frequency components and attenuates (or suppresses) the high-frequency components. This can be accomplished by applying spatial masks directly on the image.

Spatial filters are one of the most important tools in image processing. The filters in image processing can be characterized into the following three categories:

1. Convolution-based filters
2. Order-statistics (rank) filters
3. Hybrid filters

Convolution-based filters use spatial masks that are also known as kernels, templates, or windows. A spatial mask is convolved with the given image to achieve the required smoothing or sharpening effect. *Order-statistics (rank/rank-order/order) filters* do not use any convolution techniques. They simply arrange the pixels that are under the mask in a desired order. Then rank is chosen as the resultant. For example, for the median filter, the middle value is chosen as the resultant. *Hybrid filters* use the concepts of both ranking and convolution. Unsharp masking is a good example of hybrid filters. Image smoothing uses convolution-based filters. Rank filters are mainly used for image restoration and will be dealt with in Chapter 6.

A spatial filter is characterized by the following two aspects:

1. Concept of neighbourhood
2. Convolution operation over the neighbourhood

Spatial filters can further be classified as either linear or non-linear spatial filters, based on the nature of response. For a linear filter, the output is a linear combination of input pixels. This is not the case in a non-linear filter.

A sample 3 × 3 spatial mask is shown in Fig. 5.31, where w_1–w_9 are mask weights.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Fig. 5.31 Sample 3 × 3 spatial filter mask

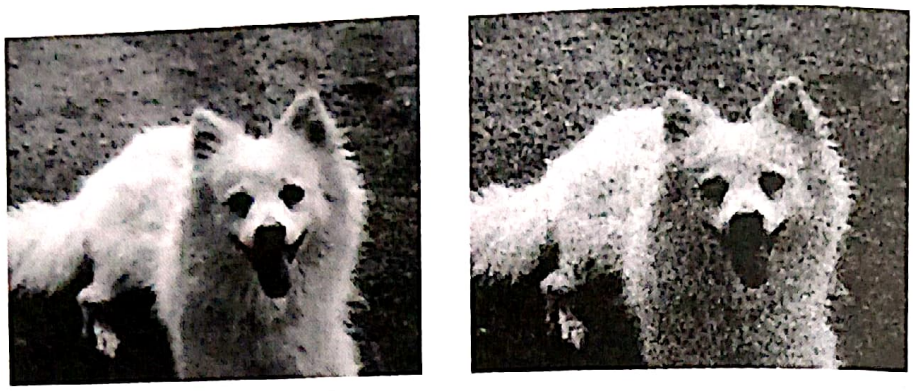
As per the convolution operation, the mask is placed over the image and the convolution process is applied. The image mask is shown in Fig. 5.32. Now, the filtering process can be described as the convolution process of the image and the spatial filter mask.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Fig. 5.32 Sample 3 × 3 image

5.6 IMAGE SMOOTHING SPATIAL FILTERS

A smoothing filter is a linear filter that creates an image with a smooth appearance by blurring the image and removing noise (e.g., Gaussian noise). Noise is an unnecessary disturbance in images. One such noise is Gaussian noise.



(a) (b)

Fig. 5.33 Illustration of Gaussian noise (a) Original image (b) Image with Gaussian noise

The original image and the Gaussian noise-affected image are shown in Figs 5.33(a) and 5.33(b), respectively.

5.6.1 Box Filters

Image smoothing is done by a mean filter. The averaging process eliminates the extreme values of a neighbourhood in the spatial domain. A simple 1D box filter can be designed as follows:

$$\frac{1}{3}[1\ 1\ 1]$$

The idea is to add all the pixels of the image within this spatial mask and to divide the sum by the number of pixels. Averaging reduces the pixel intensity of the neighbourhood, thus leading to smoothness. This 1D filter can be extended to a 2D box filter as follows:

$$\frac{1}{3}[1\ 1\ 1]\times\frac{1}{3}\begin{bmatrix}1\\1\\1\end{bmatrix}$$

This leads to a 2D spatial mask, as shown in Fig. 5.34. This shows that a 2D filter is separable, and it is possible to decompose it into 1D horizontal and vertical components.

$$\frac{1}{9}\times\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

Fig. 5.34 Sample 3 × 3 2D spatial mask

The filtering process is carried out by convolving this spatial mask and the image. This process of spatial averaging is described as follows:

$$g(x,y)=\frac{1}{N}\sum_{(x,y)\in N}^{n} f(x,y)$$

where N is the neighbourhood. For the aforementioned 3 × 3 spatial mask, there will be nine pixels in the neighbourhood. Thus, the value of the centre pixel is replaced by the average of all the values in the neighbourhood.

Example 5.14 If an image is given as

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

what would be the output of a box filter?

Solution The application of a 3×3 box filter results in

$$\frac{1}{9}[1+2+3+5+4+6+7+8+9] = \frac{1}{9}[45] = 5.$$

This would replace the centre pixel. In other words, the value of the centre pixel, which is 4, will now be replaced by 5.

Spatial masks can be extended to any size, such as 5×5, 7×7, and 9×9, using the same logic used for creating 3×3 spatial masks. In general, odd-numbered masks are used as they have a clear central value. This kind of a spatial filter is called a box filter. A filter with a large spatial mask results in the loss of more image details. This is illustrated in Figs 5.35(a)–(d).

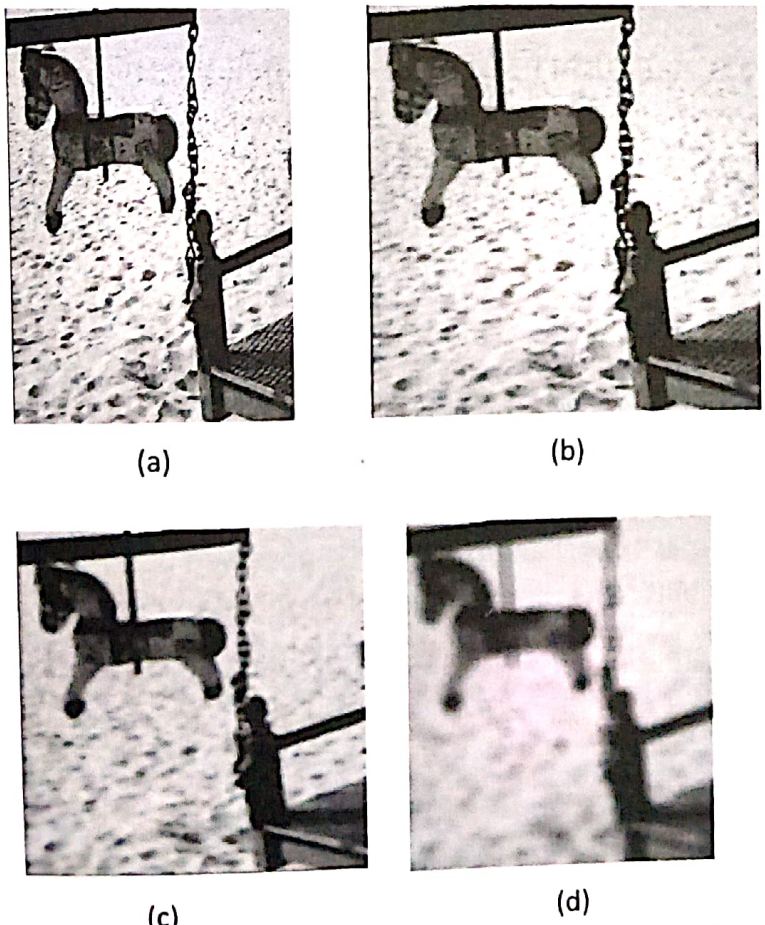


Fig. 5.35 Box filters (a) Original image (b) Filtering with 3×3 mask (c) Filtering with 5×5 mask (d) Filtering with 11×11 mask

Similarly, shapes of the masks can also be changed. Some of the mask shapes are shown in Fig. 5.36.

$$\begin{pmatrix} & X & \\ X & X & X \\ & X & \end{pmatrix}, \begin{pmatrix} X & X & X \\ X & X & X \\ X & X & X \end{pmatrix}, \begin{pmatrix} & & X & & \\ & X & X & X & \\ X & X & X & X & X \\ & X & X & X & \\ & & X & & \end{pmatrix}$$

Fig. 5.36 Different kinds of masks

5.6.2 Gaussian Filters

To reduce the loss of visual information, it is possible to assign different weights to mask coefficients so that more importance is attached to some pixels over others. This is to ensure that the filter has one peak. Suppose $f(x-1)$, $f(x)$, and $f(x+1)$ represent three points. Let us assign a higher weight for the middle (1/2) and lower weights for the other pixels indicated, as follows:

$$f(x)=\frac{1}{4}f(x-1)+\frac{1}{2}f(x)+\frac{1}{4}f(x+1)$$

This equation is equivalent to applying the following 1D mask:

1/4	1/2	1/4

The following 2D spatial mask can be constructed from this 1D mask:

$$\frac{1}{4}[1 \quad 2 \quad 1]\times\frac{1}{4}\begin{bmatrix}1\\2\\1\end{bmatrix}$$

The resultant mask is shown in Fig. 5.37.

Often, these types of masks are multiplied by $1/N$, where N is the sum of the mask coefficients. It can be observed that the coefficients are biased as more weight is assigned to the centre pixels. Many such application-dependent masks can be designed, as shown in Fig. 5.38.

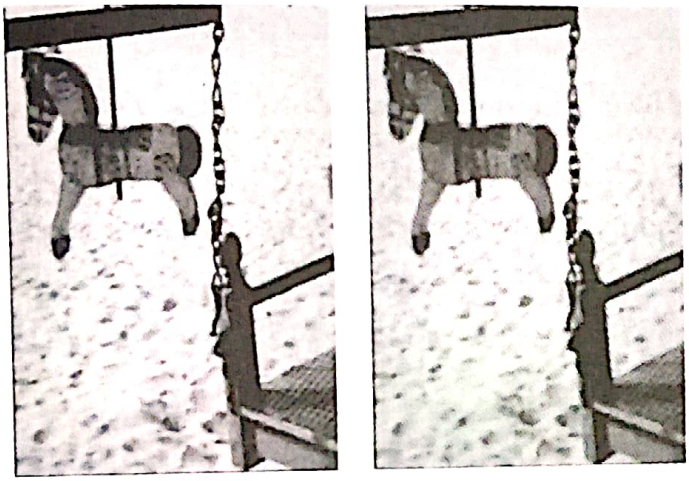
$$\frac{1}{16}\times\begin{array}{|c|c|c|}\hline 1&2&1\\\hline 2&4&2\\\hline 1&2&1\\\hline\end{array}$$

Fig. 5.37 Sample mask with variable weights

$$\frac{1}{10}\begin{pmatrix}1&1&1\\1&2&1\\1&1&1\end{pmatrix} \quad \frac{1}{4}\begin{pmatrix}0&1&0\\1&0&1\\0&1&0\end{pmatrix} \quad \frac{1}{8}\begin{pmatrix}1&1&1\\1&0&1\\1&1&1\end{pmatrix}$$

Fig. 5.38 Sample masks with variable weights

Gaussian filters are an approximation of the aforementioned variable weight filters. Gaussian filters choose the weights of the mask according to the shape of a Gaussian function. The value of the centre pixel is large, and as the distance between the pixel and centre increases, the mask weight decreases. Hence, Gaussian filters are very useful in image smoothing as well as in removing noises of normal distribution types. The results of applying Gaussian filters are shown in Figs 5.39(a) and (b).



(a) (b)

Fig. 5.39 Gaussian filters (a) Filtering with low-pass 3 ×3 filter (b) Filtering with low-pass 5 × 5 filter

5.6.2.1 Design of Discrete Gaussian Masks

Gaussian filters can be designed using two methods. The zero mean Gaussian function in 1D is given as follows:

$$G(x) = e^{-\left(\frac{x^2}{2\sigma^2}\right)}$$

Here, σ is the width of the Gaussian function. The 2D Gaussian mask has values that approximate the continuous function

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

The first way of designing a Gaussian filter is to approximate a Gaussian function using a binomial expansion. In other words, the row n of a Pascal triangle serves as a 1D approximation of a Gaussian filter. The binomial coefficient is calculated as follows:

$$\binom{n}{k} = \frac{k!}{k!(n-k)!}$$

The variable k is the kth entry in row $n + 1$ of the Pascal triangle. The Pascal triangle is shown in Fig. 5.40.

		1			$\sigma = 0$
	1		1		$\sigma = 0 + \frac{1}{4} = \frac{1}{4}$
1		2		1	$\sigma = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$
1	3		3	1	$\sigma = \frac{1}{2} + \frac{1}{4} = \frac{3}{4}$

Fig. 5.40 Pascal triangle

For example, to form a binomial operator of size 4, one can pick the fourth row, that is,

$$[1\ 3\ 3\ 1]$$

It can be observed that the value of σ differs by 1/4 for each row. Now add the contents of the row and divide each element by 8 to provide a binomial 1D mask as follows:

$$\left[\frac{1}{8}\ \frac{3}{8}\ \frac{3}{8}\ \frac{1}{8}\right]$$

A 2D Gaussian mask can be created from a 1D mask using the same logic used earlier.

If the mask size is $2n + 1$, then σ and the order n are related by the formula $n = 2\sigma + 1$. Thus, when $\sigma = 1$, the mask size is 7 and when $\sigma = 2$, the mask size is 11.

The second way of designing a Gaussian mask is to compute the weights directly using the following formula:

$$G(x, y) = c \times e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$

The variable c is called the normalizing constant. Choosing a value of σ and substituting the values in the exponential equation gives the mask weights in fractions. However, for mask weights, large positive numbers are preferred. Therefore, by choosing an appropriate value for c, fractional weights can be converted to positive numbers.

5.6.3 Directional Smoothing

Most of the filters are isotropic since their effect is same in all directions. In image processing applications, it may sometimes be necessary to select only certain features, in particular directions, such as edges along horizontal, vertical, or diagonal directions. This kind of a filter is called an anisotropic filter. *Directional smoothing filters* are useful in reducing blurring of edges by excessive smoothing process. The procedure for implementing a directional filter is as follows:

1. The spatial average is calculated in several directions.
2. The direction that is associated with the minimum is detected and used as a part of the convolution process to replace the centre pixel.

Another form of selective smoothing is called *conservative smoothing*. The algorithm for conservative smoothing is illustrated as follows:

1. For a centre pixel, find the pixel values of its eight-neighbourhood.
2. Find the maximum and minimum pixel values.
3. Compare the value of the centre pixel with the maximum and minimum values.
   (a) If the value of the centre pixel > maximum value, set the centre pixel value to the maximum value.
   (b) If the value of the centre pixel < minimum value, set the centre pixel value to the minimum value.
   (c) Otherwise, retain the centre pixel value as it is.

Then the mask is moved and the process is repeated till the entire image is processed. This algorithm retains most of the important visual information. The original image is shown in Fig. 5.41(a). Gaussian noise is added to it and the resulting image is shown in Fig. 5.41(b). A conservative filter is applied and the resultant is shown in Fig. 5.41(c).

(a) (b) (c)

Fig. 5.41 Conservative filters (a) Original image (b) Image with Gaussian noise (c) Output with conservative filter

5.7 IMAGE SHARPENING SPATIAL FILTERS

Image sharpening filters highlight the details of an image. High-spatial-frequency components have detailed information. This is in the form of edges and boundaries. Edges are significant local intensity variations that exist between two different regions. As discussed in Chapter 2, edges provide the outlines of objects. Human visual system also performs some sort of edge detection task at various levels for recognizing the objects. Image sharpening algorithms are used to separate object outlines. Therefore, these filters are also known as *edge enhancement* or *edge crispening* algorithms.

5.7.1 Gradient and Laplacian Filters

Edges can be extracted by taking the gradient of the image. Gradient refers to the difference between the pixels of an image. If the neighbouring pixels have the same intensity, the difference is zero and hence there is no edge, as edges exist only at points where there is

a significant local intensity variation. However, if there is a difference in the intensities between two neighbouring pixels, it indicates the presence of an edge point. By comparing this with a threshold, one can identify whether this is a significant edge point or not.

As images are two dimensional, the gradient vector of an image $f(x, y)$ 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:

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

or simply

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

where $g_x = \left[\dfrac{\partial f(x,y)}{\partial x}\right]$ and $g_y = \left[\dfrac{\partial f(x,y)}{\partial y}\right]$, and the operator ∇ is called nabla or gradient operator.

The magnitude of this vector (or length of the gradient or norm) is given as follows:

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

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:

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

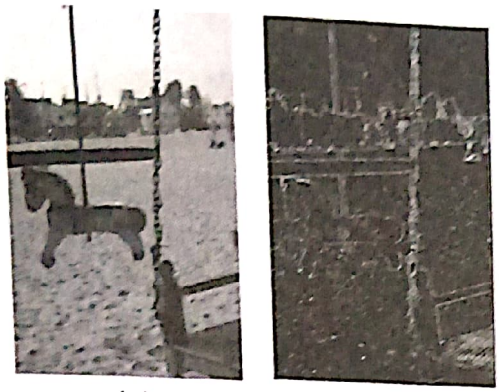
or

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

The gradient direction can be given as follows:

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

An original image and its edges extracted using a first-order derivative are shown in Figs 5.42(a) and (b), respectively.



(a) (b)

Fig. 5.42 Edge detection (a) Original image (b) Extracted edges

Edge detection algorithms are discussed in Chapter 9 as segmentation of edges. In this section, only filters that process the high-frequency components like edges are dealt with.

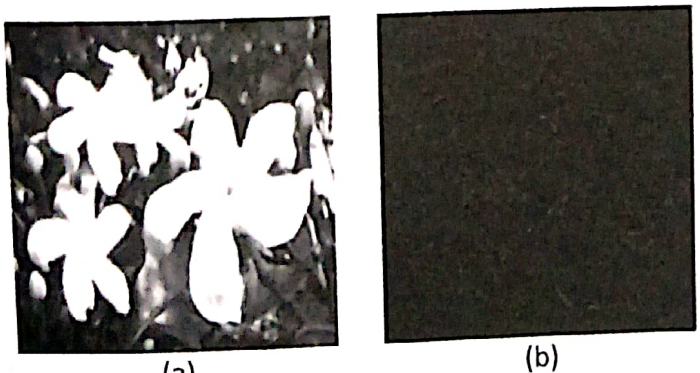
The Laplacian is a second-order gradient of the image, and its masks are given in Fig. 5.43. Any one of the masks shown can be used.

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & +4 & -1 \\ 0 & -1 & 0 \end{pmatrix} \qquad \begin{pmatrix} -1 & -1 & -1 \\ -1 & -8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Fig. 5.43 Four sample Laplacian masks

Convolving the mask with an image gives the output at points where the edges are highlighted. The filter also removes all the low-frequency components like the contrast in the image. One sample output of the Laplacian filter is shown in Fig. 5.44(b). It can be observed that the image is not clear. Hence, high-pass filtering requires some sort of post-processing steps like histogram equalization to ensure that the image is of acceptable quality.



(a) (b)

Fig. 5.44 Image sharpening spatial filters (a) Original image (b) Laplacian high-pass filter result

A high-pass filter can be implemented using a Laplacian mask. A high-pass filter removes all the low-frequency components, that is, all contrast information is lost. This affects the visual quality of the image. However, high-pass filters retain all high-frequency information related to edges and boundaries. Image enhancement of the image $f(x, y)$ can be done by adding the Laplacian edge result, which enhances the high-frequency contents and thereby gives an edge-enhanced image. This is expressed mathematically as follows:

$$g(x,y) = f(x,y) \pm \nabla^2 f(x,y)$$

(a) (b) (c)

Fig. 5.45 Image sharpening spatial filter (a) Original image (b) Result of Laplacian high-pass filter (c) Enhanced image

Figure 5.45(a) shows the original image that needs to be enhanced. Figure 5.45(b) shows the Laplacian output that is added to obtain the edge-enhanced image in Fig. 5.45(c).

5.7.2 High-boost Filters

A high-boost filter is used to prevent the loss of low-frequency components and restore the visual details of an image. This is achieved by adding an offset to the filtered image, thereby retaining some of the low-frequency components. A high-pass filtered image may be computed as the difference between the original image and a low-pass filtered version of that image, as follows:

(High-pass part of image) = (original image) − (low-pass part of image)

Multiplying the original image by an amplification factor denoted by A yields the high-boost filter.

High-boost image = (A) (original) − (low pass)
= $(A - 1)$ (original) + (original − low pass)
= $(A - 1)$ (original) + (high pass)

The mask that implements this procedure is shown in Fig. 5.46(a). The result of the filter is shown in Fig. 5.46(b). It can be observed that the result of a high-boost filter is better than that of a high-pass filter.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & A & 0 \\ 0 & 0 & 0 \end{bmatrix} + \frac{1}{9} \times \begin{bmatrix} -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

(a)

(b)

Fig. 5.46 Image sharpening spatial filter (a) Spatial mask for high-boost filter (b) Result of high-boost filter

The high-boost filtered image looks more like the original image with a degree of edge enhancement, depending on the value of A. Any $N \times N$ high-boost filter mask can be created using the value of $A = (N \times N) - 1$. If a mask of size 5×5 is used, then the value of A will be 24. Since the preceding mask is 3×3, the ideal value of A is 8. If $A = 1$, this mask results in a high-pass filter. In that case, only the edges will be present in the image. If the value is less than 8, the image will appear as the negative of the original image. This mask can be extended for higher orders, say 7×7 or 13×13, by making changes in the weight and gain A. Larger masks will emphasize the edges more and help mitigate the effects of noise (if present) in the original image.

5.7.3 Unsharp Masking

Unsharp masking is a hybrid filter technique. This filter is very useful for removing both impulse and Gaussian noise present in an image. An unsharp filter is a simple sharpening operator, which derives its name from the fact that it enhances edges (and the other high-frequency components) in an image, and this technique is commonly used in photographic and printing industries for this purpose.

The procedure for implementing an unsharp mask is as follows:

1. Read the image.
2. Blur the image using any image smoothing filter. This stage requires a convolution-based smoothing filter. Let the smooth or blurred image be $\overline{f}(x, y)$.
3. Let the mask = original image $- \overline{f}(x, y)$. Subtraction of the blurred version from the original image results in an image with a visible emphasis on edges.

4. Add the weighted portion of the mask to the original image, to restore some of the lost visual information. Hence,

$$g(x,y) = f(x,y) + k \times \text{mask}$$

If $k = 1$, this procedure is called unsharp masking. When $k > 1$, this process is called high-boost filtering, and when $k < 1$, this filter is used to de-emphasize the contribution of unsharp masking. Consider the original image shown in Fig. 5.47(a). An image smoothing filter like Gaussian filter is applied to the original image and its result is subtracted from the original image. The difference image is shown in Fig. 5.47(b). The resultant images obtained with different values of k are shown in Figs 5.47(c)–(f). It can be observed that the contrast of the image is much better than that in the case of a high-pass filter.

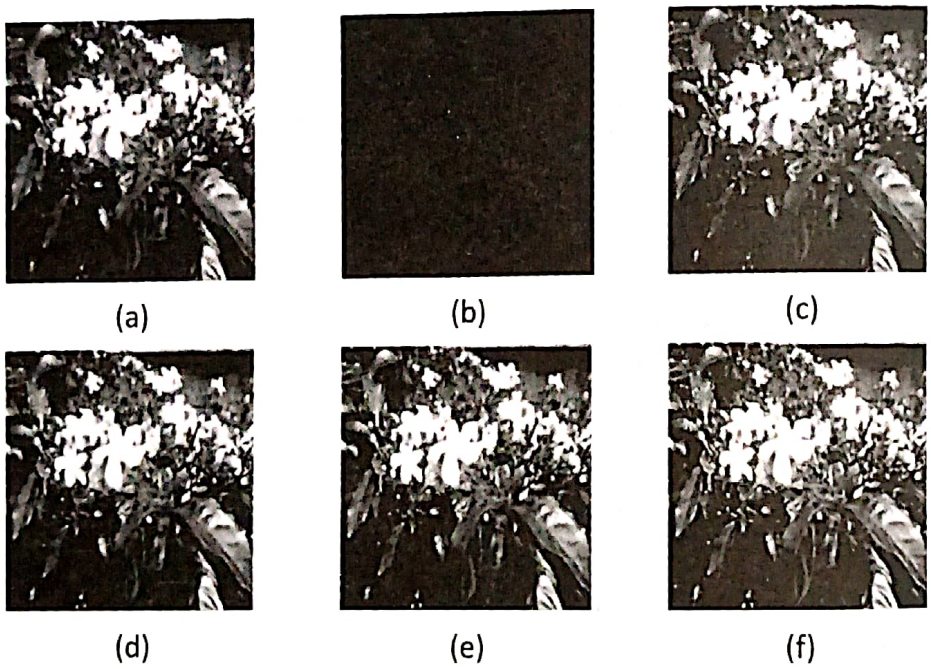


Fig. 5.47 Unsharp masking (a) Original image (b) Difference between original and Gaussian (c) Resultant image with $k = 0.3$ (d) Resultant image with $k = 0.5$ (e) Resultant image with $k = 0.7$ (f) Resultant image with $k = 2.0$

5.8 IMAGE SMOOTHING IN FREQUENCY DOMAIN FILTERING

The filtering process can also be performed in the frequency domain. In Chapter 4, we have discussed how to convert spatial-domain information to frequency-domain information using Fourier transforms. The frequency-domain filtering process is based on image transforms. The greatest advantage of the frequency domain is that one can manipulate the spatial frequency components independently. A sample 2D low-frequency mask is shown in Fig. 5.48(c). The original image is shown in Fig. 5.48(a). The Fourier transform of the image, mask for the low-pass filter, and resultant image of the low-pass filter are shown in Figs 5.48(b), (c), and (d), respectively. Frequency-domain masks can be designed depending

on the application requirement. Hence, by controlling the weights of the frequency-domain masks, one can control the attenuation of the frequency components and indirectly the nature of image enhancement. Frequency domain filtering is preferred to spatial domain filtering because of fewer computations involved. This is because convolution in the spatial domain is equivalent to multiplication in the frequency domain. For smaller masks (up to 9 × 9), the spatial domain is effective, but for larger masks, filtering in the frequency domain is preferred.

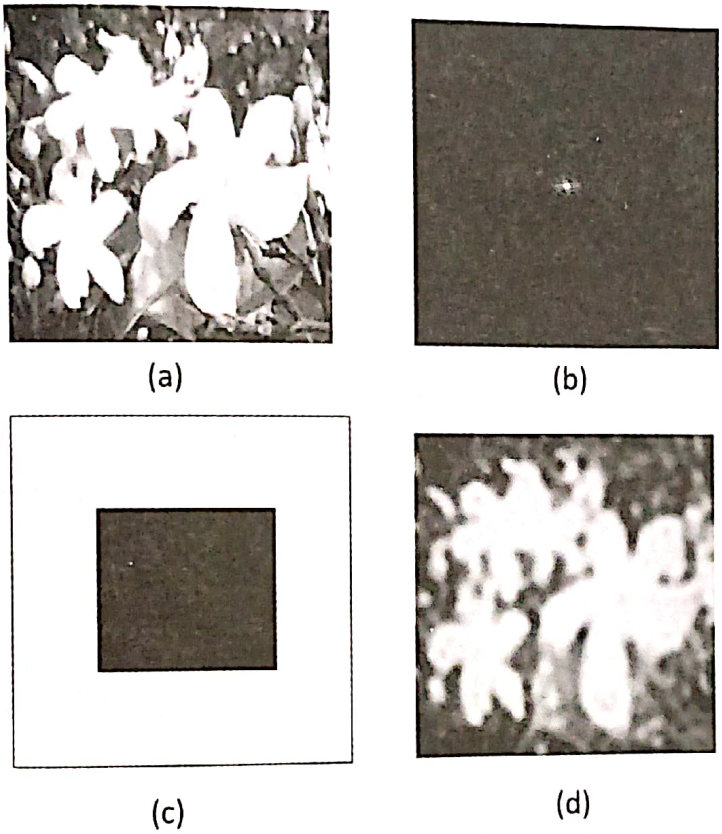


Fig. 5.48 Frequency domain low-pass filtering (a) Original image (b) Fourier transform (c) Low-pass circular mask (d) Result of low-pass filter

The algorithm for frequency domain filtering is given as follows:

1. Let $f(x, y)$ be the original image for which filtering is required. Obtain the Fourier transform of the image. Read the spectrum and multiply by $(-1)^{x+y}$ to centre the transform.
2. Design a frequency-domain filter matrix function $h(x, y)$. The mask shape can be a circle, rectangle, or any custom shape depending on the application requirement. Obtain the Fourier spectrum of $h(x, y)$ to get $H(u, v)$.
3. Multiply the Fourier spectrum of the filter with the Fourier spectrum of the image by element-wise multiplication, as discussed in Chapter 3:

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

where $G(u, v)$ is the filtered output in the frequency domain.

on the application requirement. Hence, by controlling the weights of the frequency-domain masks, one can control the attenuation of the frequency components and indirectly the nature of image enhancement. Frequency domain filtering is preferred to spatial domain filtering because of fewer computations involved. This is because convolution in the spatial domain is equivalent to multiplication in the frequency domain. For smaller masks (up to 9 × 9), the spatial domain is effective, but for larger masks, filtering in the frequency domain is preferred.

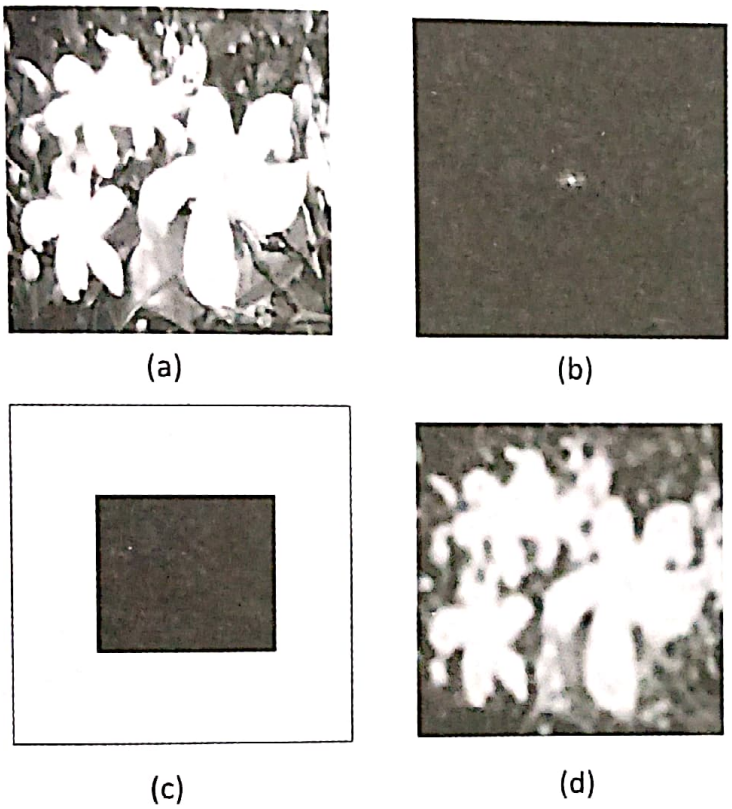


Fig. 5.48 Frequency domain low-pass filtering (a) Original image (b) Fourier transform (c) Low-pass circular mask (d) Result of low-pass filter

The algorithm for frequency domain filtering is given as follows:

1. Let $f(x, y)$ be the original image for which filtering is required. Obtain the Fourier transform of the image. Read the spectrum and multiply by $(-1)^{x+y}$ to centre the transform.
2. Design a frequency-domain filter matrix function $h(x, y)$. The mask shape can be a circle, rectangle, or any custom shape depending on the application requirement. Obtain the Fourier spectrum of $h(x, y)$ to get $H(u, v)$.
3. Multiply the Fourier spectrum of the filter with the Fourier spectrum of the image by element-wise multiplication, as discussed in Chapter 3:

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

where $G(u, v)$ is the filtered output in the frequency domain.

4. Apply the inverse transform to $G(u, v)$ to retrieve the filtered image in the spatial domain.
5. Extract the real components of the resultant and multiply by $(-1)^{x+y}$ to offset the effect of step 1.
6. Display the images and exit.

This algorithm is general. It is used to implement many frequency-domain filters. Let us now discuss some of the important frequency-domain filters.

5.8.1 One-dimensional Ideal Low-pass Filters

An ideal low-pass filter allows all frequencies up to a certain cut-off frequency D_0 and removes all frequencies beyond that. Its transfer function is given as follows:

$$H(D) = \begin{cases} 1 & \text{for } D \le D_0 \\ 0 & \text{for } D > D_0 \end{cases}$$

Multiplication of this with the image $F(u)$ in one dimension preserves the frequencies up to D_0. This is equivalent to low-pass filters. All the frequencies greater than D_0 are removed. This process gives a smooth effect to the image.

5.8.2 Two-dimensional Ideal Low-pass Filters

Images are two dimensional. Hence, $H(D)$ should be applied first along the rows of the image and the results should be stored in an intermediate image. Then, $H(D)$ should be applied to the columns of the intermediate image to yield a 2D mask. Alternatively, ideal 2D mask $H(u, v)$ can be designed as follows:

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

However, a more effective approach is to use a single filter and apply it radially on all the frequencies of the image. The masks can be rectangular, circular, or of any custom shape. First, the radial distance of the point (u, v) from the centre of the frequency rectangle is calculated and described as follows:

$$D(u,v) = \left[\left(u - \frac{M}{2}\right)^2 + \left(v - \frac{N}{2}\right)^2\right]^{\frac{1}{2}}$$

where $D(u, v)$ is the radial frequency, which involves square root because the origin of the image (whose size is $M \times N$) is entered in the frequency rectangle. Now,

$$(u, v) = \left[\frac{M}{2}, \frac{N}{2}\right]$$

Now for two dimension, radially implemented filter D_0 is the radial cut-off frequency and is a positive constant. In a circular mask, the cut-off frequency is the radius of the circle. The unit of cut-off frequency is specified in terms of pixels. Depending on the application, it has to be selected carefully.

(a) (b)

Fig. 5.49 Frequency domain low-pass-filter (a) Original image (b) Result of ideal low-pass filter (with $D_0 = 20$)

The original image is shown in Fig. 5.49(a) and the resultant image of the low-pass filter with a cut-off frequency of 20 is shown in Fig. 5.49(b).

5.8.3 Butterworth Low-pass Filters

Low-pass Butterworth filters are also effective low-pass filters. These filters reduce or eliminate ringing artefacts. The profile or transfer function of the Butterworth filter is described as follows:

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

Here, n is the order of the filter and D_0 is the cut-off frequency. H represents the filter mask magnitude, the values of which range from 0 to 1. If n increases, the filter becomes sharper with increased ringing in the spatial domain. If n is 1, it produces no ringing effect and if n is 2, ringing is present but imperceptible.

(a) (b)

Fig. 5.50 Butterworth low-pass filter (a) Original image (b) Result of low-pass Butterworth filter (with $D_0 = 20$)

The original image is shown in Fig. 5.50(a). The resultant image of the low-pass Butterworth filter with a cut-off frequency of 20 is shown in Fig. 5.50(b).

5.8.4 Gaussian Low-pass Filters

The problem with an ideal low-pass filter is that it produces ringing artefacts. This is a visual effect where a series of lines of decreasing intensity lie parallel to the edges. The ringing artefacts are also called Gibbs ringing. To avoid these artefacts, Gaussian low-pass or Butterworth low-pass filters are preferred, as these filters do not have a steep profile similar to the ideal filter.

The profile of a 2D Gaussian filter is given as follows:

$$H(u,v) = e^{-\left(\frac{D^2(u,v)}{2D_0^2}\right)}$$

Here, $H(u, v)$ is the magnitude of a 2D Gaussian mask. The values of the mask range from 0 to 1. D_0 is the cut-off frequency. The Gaussian profile is controlled by the value of σ. A decrease or increase in σ causes a similar decrease or increase in the cut-off frequency. This can easily be extended to 2D images. One advantage of a Gaussian filter is that it never causes ringing effect.

(a) (b)

Fig. 5.51 Gaussian low-pass filter (a) Original image (b) Result of low-pass Gaussian filter (with $D_0 = 20$)

The original image is shown in Fig. 5.51(a) and the resultant image of the low-pass Gaussian filter with a cut-off frequency of 20 in Fig. 5.51(b).

5.9 IMAGE SHARPENING IN FREQUENCY DOMAIN

Similar to low-pass Butterworth and Gaussian filters, high-pass filter equivalents can be designed to attenuate the low-frequency components and allow the high-frequency components, such as edges, boundaries, and other abrupt changes in an image.

5.9.1 Ideal High-pass Filters

A high-pass filter passes all frequencies higher than a certain cut-off frequency D_0 and removes all frequencies below that. It is the complement of a low-pass filter. The transfer function for a high-pass filter can be described by the following relation:

$$H(D) = \begin{cases} 1 & \text{for } D \le D_0 \\ 0 & \text{for } D > D_0 \end{cases}$$

5.9.2 Ideal 2D High-pass Filters

Similarly, the transfer function for a 2D high-pass filter can be given as follows:

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

The application of a high-pass filter is shown in Figs 5.52(a)–(d).

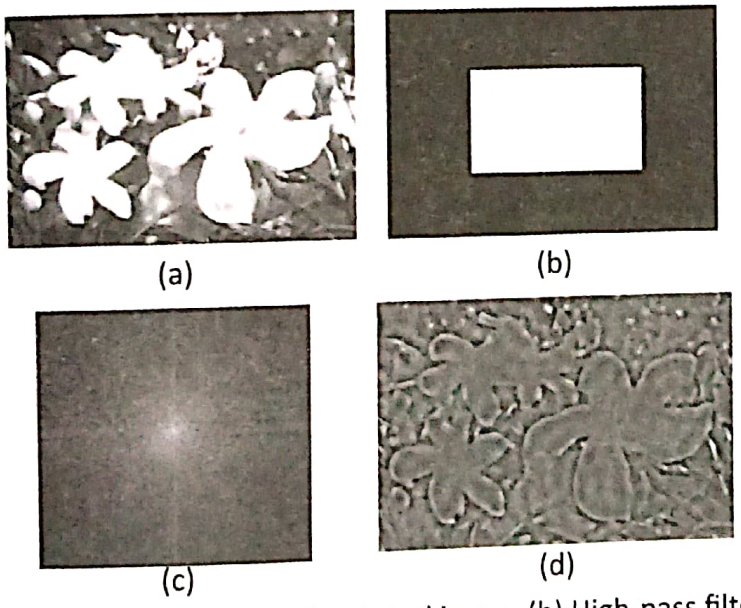


Fig. 5.52 High-pass filter (a) Original image (b) High-pass filter mask (c) Fourier transform of original image (d) Result of high-pass filter

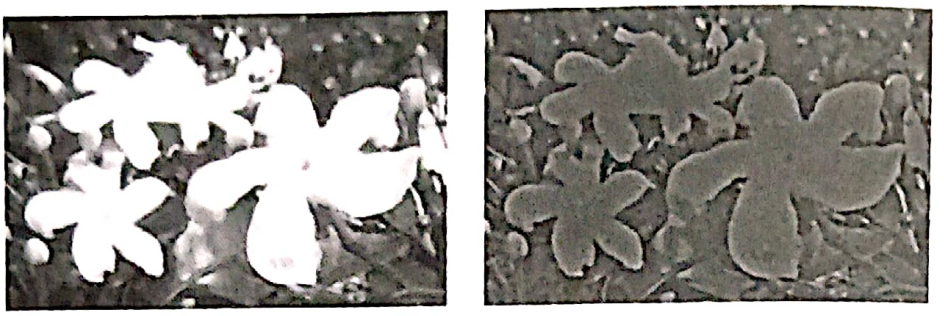
High-pass filters eliminate the zero-frequency (DC) components in an image, which results in unwanted ringing.

5.9.3 Butterworth High-pass Filters

Similarly, in the transfer function a high-pass Butterworth filter is given as follows:

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

The order n determines the sharpness of the cut-off value and the amount of ringing.



(a) (b)

Fig. 5.53 Butterworth high-pass filter (a) Original image (b) Result of high-pass Butterworth filter ($D_0 = 20$)

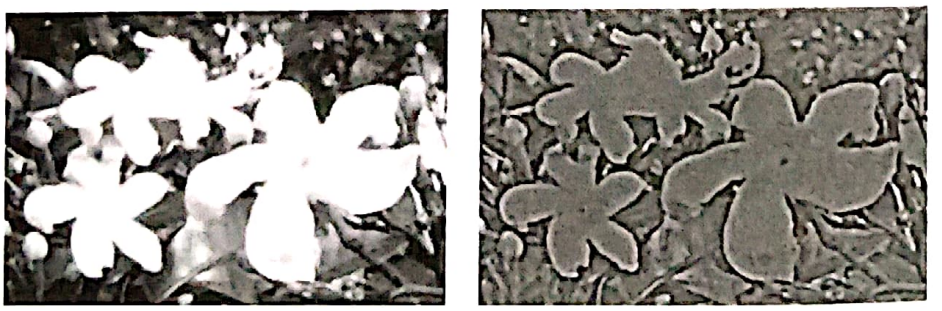
The original image is shown in Fig. 5.53(a) and the resultant image of the high-pass Butterworth filter with a cut-off frequency of 20 in Fig. 5.53(b).

5.9.4 Gaussian High-pass Filters

The transfer function for a high-pass Gaussian filter is given as follows:

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

The application of a Gaussian high-pass filter on the original image shown in Fig. 5.54(a) is illustrated in Fig. 5.54(b).



(a) (b)

Fig. 5.54 Gaussian high-pass filter (a) Original image (b) Result of high-pass Gaussian filter ($D_0 = 20$)

5.9.5 High-frequency Emphasis Filters

Another related useful filter is the frequency emphasis filter. A frequency emphasis filter emphasizes frequencies by adding a portion of the high frequencies to the image. It can be obtained by applying the inverse Fourier transform as follows:

$$g(x,y) = \text{IFFT}\{[1 + k[1 - H_{\text{LP}}(u,v)]]F(u,v)\}$$

IFFT is the inverse Fast Fourier transform. H_{LP} is the transfer function of a low-pass filter, and it can be noted that $1 - k \times H_{LP}(u, v)$ gives the transfer function of the high-pass filter. Therefore, the preceding expression is called a high-frequency emphasis filter. Here, the parameter k controls the proportion of high frequencies in the image. The preceding expression can also be generalized as follows:

$$g(x,y) = \text{IFFT}\{[(k_1 + k_2)H_{LP}(u,v)]F(u,v)\}$$

The parameter k_1 controls the offset from the origin and k_2 controls the contribution of high frequencies.

(a) (b) (c)

Fig. 5.55 High-frequency emphasis filter (a) Original image (b) Result of high-boost filter (c) Result of high-frequency emphasis filter

The original image is shown in Fig. 5.55(a). The result of a high-frequency emphasis filter is shown in Fig. 5.55(c), which can be compared with the result of the high-boost filter shown in Fig. 5.54(b).

5.10 HOMOMORPHIC FILTERING

In the simple image model described in Chapter 1, we have discussed how light from a source illuminates an object and reflections from the object are captured by sensors to form an image. Thus, an image $f(x, y)$ is considered as a product of the illumination and reflectance components. The illumination component has the low-frequency components of the image, whereas the reflectance component is associated with the high-frequency components of the image. The idea of a homomorphic filter is to separate these components and apply two different transfer functions to have more control. For example, it may be required to enhance the high-frequency components and reduce the number of frequency components associated with the illumination component. If $L(x, y)$ is the illumination component and $R(x, y)$ the reflectance component, then the image $f(x, y)$ can be represented as follows:

$$f(x, y) = L(x, y) \times R(x, y)$$

The main issue now is separation of these two components. Now, the problem with Fourier transform is that

$$\text{Fourier transform (FT) } \{A \times B\} \neq \text{FT}\{A\} + \text{FT}\{B\}$$

However,

$$\text{Fourier transform (FT) } \{A + B\} = \text{FT}\{A\} + \text{FT}\{B\}$$

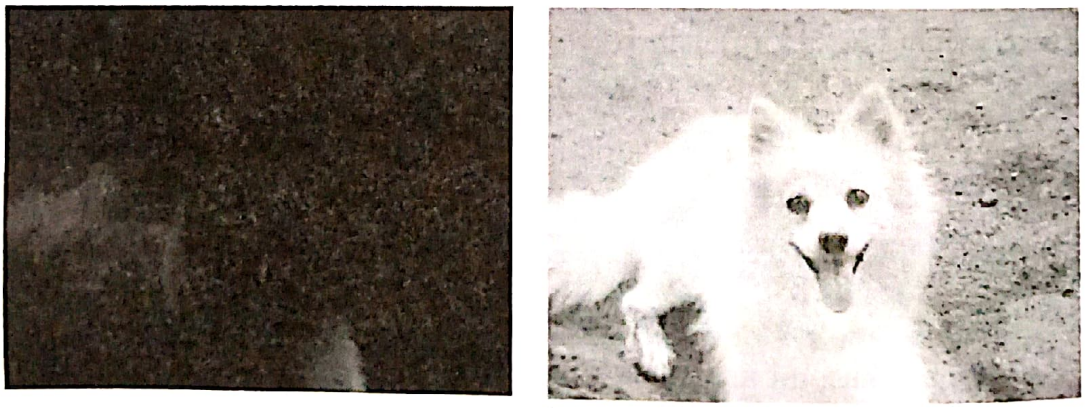
Therefore, logarithm (log function) is first applied to the image so that it can be expressed as a sum of its illumination and reflectance components. The procedure for applying the homomorphic filter is as follows:

1. Apply log transformation to the image. This results in the following expression:

$$\ln[F(x,y)] = \ln[L(x,y)] + \ln[R(x,y)]$$

2. Apply Fourier transform to the log of these components.
3. Design filters separately for the illumination and reflectance components. The transfer function of the filters can be different for the two components.
4. Apply the inverse Fourier transform to the filtered image.
5. To offset the logarithm applied in step 1, apply anti-log function.

The original image is shown in Fig. 5.56(a) and the resultant image obtained using a homomorphic filter is shown in Fig. 5.56(b).



(a) (b)

Fig. 5.56 Homomorphic filter (a) Original image (b) Image obtained using homomorphic filter

This chapter focused on image enhancement algorithms. As discussed earlier, these algorithms are subjective in nature and use a trial-and-error process. As images can be degraded by noise, blur, and artefacts, which pose severe challenges, these need to be removed by modelling and other sophisticated algorithms. These algorithms are discussed in the next chapter.

SUMMARY

- Images are affected by noise, blur, unnatural colours, and artefacts.
- Histograms can help understand the distribution of pixels of an image and can serve as an image quality assessment tool. A histogram is a useful summary of the distribution of grey levels. It is required for an assessment of the quality of the image.
- The purpose of image enhancement is to enhance the quality of an image by manipulating its brightness and contrast.
- Intensity transformations can be linear or non-linear functions that operate on given image pixels to improve quality.
- Histogram stretching is used to spread the histogram to cover the entire dynamic range without changing the shape of the histogram.
- Histogram equalization tries to flatten the histogram to create a better-quality image. Histogram specification allows more control over the histogram process.
- Objective fidelity criteria use metrics to quantify the error for quality assessment. Subjective quality assesses images based on the subjective opinion of human observers.
- The signal-to-noise ratio is a useful metric used for quality assessment. The value of the SNR is very high in good-quality images. The contrast-to-noise ratio can also be used to assess the quality of the image.
- Contrast stretching is a piecewise linear function that is used to manipulate the contrast of an image.
- Grey-level or intensity slicing is used to highlight a specific range of intensity levels in an image.
- Bit-plane slicing is used to transform a grey-level image into a sequence of binary images by taking advantage of its bit pattern storage.
- Image enhancement can also be performed in the frequency domain.
- Image smoothing is useful for giving a softer visual effect to an image and also for removing noise. Typically, ideal low-pass filters, Butterworth filters, and Gaussian filters are used for smoothing purposes.
- Image sharpening filters highlight the finer details in an image. The ideal high-pass filter and its variants are used for image sharpening.
- Unsharp masking is used to sharpen an image by subtracting the unsharp (or smooth) image from the original one.

KEY TERMS

Artefacts Noises that are not random, but systematic errors

Brightness Average pixel intensity of the image

CNR Acronym for contrast-to-noise ratio; a metric used to assess image quality

Contrast Finer details of the foreground object with respect to the background

Contrast stretching Operation to manipulate the contrast of the image using linear mapping

Dynamic range Difference between the maximum and minimum pixel values

Error Difference between an image and its reference image

Filters Methods that are used to choose or reject the spatial frequency components; low-pass filters allow only the low-frequency components, high-pass filters only the high-frequency components attenuating the low-frequency components

Global operations Transformations that work on all the pixels of an image

Histogram Plot of intensity versus the number of pixels that share intensity, thus serving as a summarization tool for displaying the distribution of the grey levels of an image

Histogram normalization Operation that normalizes pixel values to their average value to improve the image quality

Image enhancement Pre-processing an image so that irrelevant information or noise can be removed and the required information is highlighted

Intensity slicing Operation that highlights a specific range of intensity levels of an image

Isotropic filter Filter that produces same edge effects in all directions

Lookup table Table that facilitates linear function mapping without affecting the original pixel values

Mask Matrix with weights for selection or rejection of frequency components in the spatial and frequency domains

Neighbourhood Group of pixels having pixel connectivity

Objective fidelity The image quality characterization using metrics such as error and SNR

Point operation Transformation that manipulates the pixels of an image

SNR Acronym for signal-to-noise ratio, which is a metric used to assess image quality

Subjective fidelity Intuitive way of assessing image quality using the human visual system

REVIEW QUESTIONS

1. What is the difference between image enhancement and image restoration?
2. What would happen if low- or high-pass filters are repeatedly applied to an image? Is this repeated operation helpful in any way?
3. What are grey-level intensity transformations?
4. What is a box filter?
5. What is the need for Gaussian low-pass filters? Design a Gaussian mask for two sigma values.
6. What is the gradient of an image? How is it helpful in image enhancement?
7. What is a high-boost filter? Show with an example. Why do we not prefer higher-order derivative filters?
8. What is unsharp masking?
9. Do frequency-domain filters have any advantage over spatial filters? Why?
10. Explain how low-pass filters can be constructed in the frequency domain.
11. What is the effect of the size and shape of a mask on the filtering process?
12. What is a homomorphic filter? What is its speciality?

NUMERICAL PROBLEMS

1. Consider the image $\begin{pmatrix} 6 & 7 & 3 \\ 5 & 2 & 4 \\ 1 & 2 & 3 \end{pmatrix}$. Perform log, square root, and exponential functions and show the results.
2. What would be the impact of removing the LSB of the pixel values in the histogram of the following image?

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

3. A given image is in the range of 10–50. To display the image on a device that has the grey-level range of 0–255, what would be the required range normalization?
4. Let the image $f(x, y)$ be $\begin{pmatrix} 1 & 3 & 5 \\ 4 & 4 & 3 \\ 5 & 2 & 2 \end{pmatrix}$ and the reconstructed image be $\hat{f}(x,y) = \begin{pmatrix} 1 & 2 & 4 \\ 4 & 4 & 2 \\ 5 & 2 & 1 \end{pmatrix}$. What are the values of MSE, SNR, and PSNR for an 8-bit image?

5. Consider the following image:

4	4	4	4	4
3	4	5	4	3
3	5	5	5	3
3	4	5	4	3
4	4	4	4	4

Apply histogram equalization to this image.

6. Perform image enhancement for the 8 × 8 image distributions shown in the following tables:

(a)

p_k	2	2	10	10	20	8	6	8
r_k	0	1	2	3	4	5	6	7

(b)

p_k	8	10	10	2	12	16	4	2
r_k	0	1	2	3	4	5	6	7

7. Perform histogram equalization for the following image:

$$\begin{pmatrix} 1 & 3 & 5 \\ 4 & 4 & 3 \\ 5 & 2 & 2 \end{pmatrix}$$

8. Perform histogram specification for the image given in Problem 5.7 to the target level shown in the following table:

P_k	0	0	0	0	2	2	4	1

9. Apply the following spatial filters to the given image $f(x, y)$ and show the intermediate results:

$$f(x, y) = \begin{pmatrix} 1 & 3 & 5 \\ 4 & 4 & 3 \\ 5 & 2 & 2 \end{pmatrix}$$

(a) Low-pass filter
(b) High-pass filter