IMAGE PROCESSING

Subject Code : 10EC763	IA Marks : 25
No. of Lecture Hrs/Week : 04	Exam Hours : 03
Total no. of Lecture Hrs. : 52	Exam Marks : 100

PART - A

UNIT - 1

DIGITAL IMAGE FUNDAMENTALS: What is Digital Image Processing. fundamental Steps in Digital Image Processing, Components of an Image processing system, elements of Visual Perception. **6 Hours**

UNIT - 2

Image Sensing and Acquisition, Image Sampling and Quantization, SomeBasic Relationships between Pixels, Linear and Nonlinear Operations. **6 Hours**

UNIT - 3

IMAGE TRANSFORMS: Two-dimensional orthogonal & unitary transforms, properties of unitary transforms, two dimensional discrete Fourier transform. **6 Hours**

UNIT - 4

Discrete cosine transform, sine transform, Hadamard transform, Haar transform, Slant transform, KL transform. **6 Hours**

PART - B

UNIT - 5

IMAGE ENHANCEMENT: Image Enhancement in Spatial domain, SomeBasic Gray Level Trans -formations, Histogram Processing, Enhancement Using Arithmetic/Logic Operations. **6 Hours**

UNIT - 6

Basics of Spatial Filtering Image enhancement in the Frequency Domain filters, Smoothing Frequency Domain filters, Sharpening Frequency Domain filters, homomorphic filtering. **6 Hours**

UNIT - 7

Model of image degradation/restoration process, noise models, Restoration in the Presence of Noise, Only-Spatial Filtering Periodic Noise Reduction by Frequency Domain Filtering, Linear Position-Invariant Degradations, inverse filtering, minimum mean square error (Weiner) Filtering **10 Hours**

UNIT - 8

Color Fundamentals. Color Models, Pseudo color Image Processing., processing basics of full color image processing **6 Hours**

TEXT BOOK:

1. **"Digital Image Processing"**, Rafael C.Gonzalez and Richard E. Woods, Pearson Education, 2001, 2nd edition.

REFERENCE BOOKS:

1. **"Fundamentals of Digital Image Processing"**, Anil K. Jain, Pearson Edun, 2001.
2. **"Digital Image Processing and Analysis"**, B. Chanda and D. Dutta Majumdar, PHI, 2003.

INDEX SHEET

Unit-1

Introduction

What Is Digital Image Processing?

An image may be defined as a two-dimensional function, f(x, y), where x and y are *spatial* (plane) coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the *intensity* or *gray level* of the image at that point. When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a *digital image*. The field of *digital image processing* refers to processing digital images by means of a digital computer. Note that a digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are referred to as *picture elements*, *image elements*, *pels*, and *pixels*. *Pixel* is the term most widely used to denote the elements of a digital image.

Fundamental Steps in Digital Image Processing

It is helpful to divide the material covered in the following chapters into the two broad categories defined in Section 1.1: methods whose input and output are images, and methods whose inputs may be images, but whose outputs are attributes extracted from those images..The diagram does not imply that every process is applied to an image. Rather, the intention is to convey an idea of all the methodologies that can be applied to images for different purposes and possibly with different objectives.
Image acquisition is the first process acquisition could be as simple as being given an image that is already in digital form. Generally, the image acquisition stage involves preprocessing, such as scaling.

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

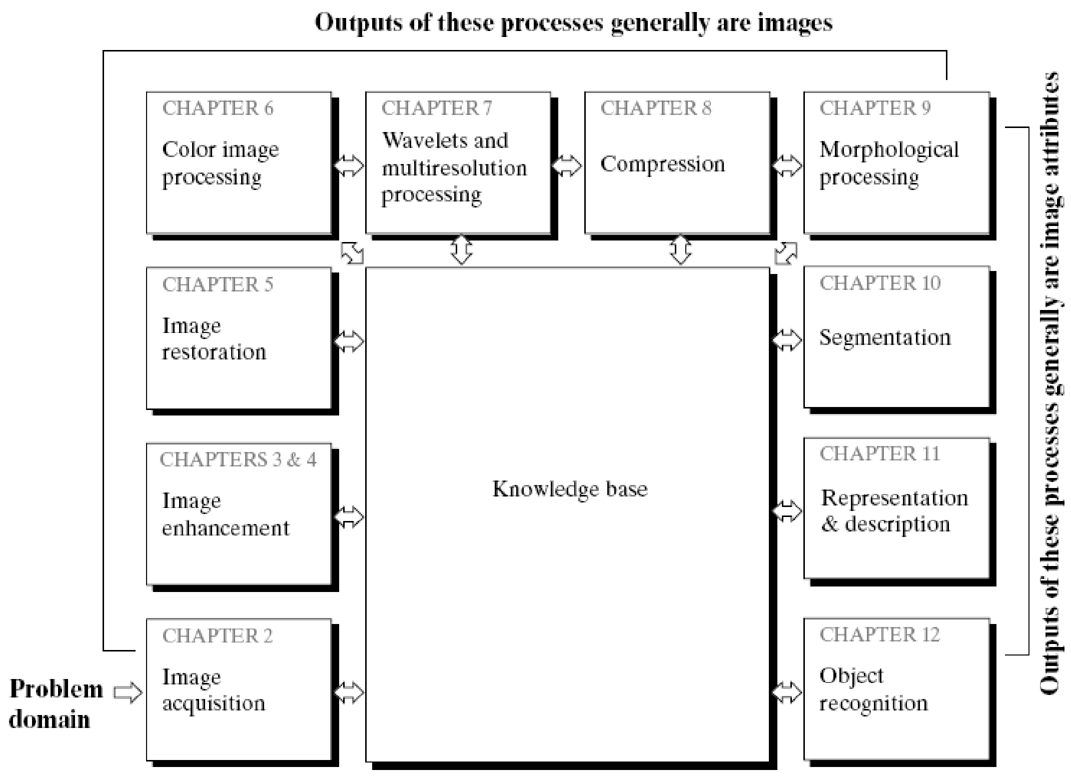


Image restoration is an area that also deals with improving the appearance of an image. However, unlike enhancement, which is subjective, image restoration is objective, in the sense that restoration techniques tend to be based on mathematical or probabilistic models of image degradation. Enhancement,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

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

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

Compression, as the name implies, deals with techniques for reducing the storage required to save an image, or the bandwidth required to transmit it. Although storage technology has improved significantly over the past decade, the same cannot be said for transmission capacity. This is true particularly in uses of the Internet, which are characterized by significant pictorial content. Image compression is familiar (perhaps inadvertently) to most users of computers in the form of image file extensions, such as the jpg file extension used in the JPEG(Joint Photographic Experts Group) image compression standard.

Morphological processing deals with tools for extracting image components that are useful in the representation and description of shape. The material in this chapter begins a transition from processes that output images to processes that output image attributes, *Segmentation* procedures partition an image into its constituent parts or objects. In general, autonomous segmentation is one of the most difficult tasks in digital image processing. A rugged segmentation procedure brings the process a long way toward successful solution of imaging problems that require objects to be identified individually. On the other hand, weak or erratic segmentation algorithms almost always guarantee eventual failure. In general, the more accurate the segmentation, the more likely recognition is to succeed.

Representation and description almost always follow the output of a segmentation stage, which usually is raw pixel data, constituting either the boundary of a region (i.e., the set of pixels separating one image region from another) or all the points in the region itself. In either case, converting the data to a form suitable for computer processing is necessary. The first decision that must be made is whether the data should be represented as a boundary or as a complete region. Boundary representation is appropriate when the focus is on external shape characteristics, such as corners and inflections. Regional representation is appropriate when the focus is on internal properties, such as texture or skeletal shape. In some applications, these representations complement each other. Choosing a representation is only part of the solution for transforming raw data into a form suitable for subsequent computer processing. A method must also be specified for describing the data so that features of interest are highlighted. *Description*, also called *feature selection*, deals with extracting attributes that result in some quantitative information of interest or are basic for differentiating one class of objects from another.

Recognition is the process that assigns a label (e.g., “vehicle”) to an object based on its descriptors. As detailed in Section 1.1, we conclude our coverage of digital image processing with the development of methods for recognition of individual objects. So far we have said nothing about the need for prior knowledge or about the interaction between the *knowledge base* and Knowledge about a problem domain is coded into an image processing system in the form of a knowledge database. This knowledge may be as simple as detailing regions of an image where the information of interest is known to be located, thus limiting the search that has to be conducted in seeking that information. The knowledge base also can be quite complex, such as an interrelated list of all major possible defects in a materials inspection problem or an image database containing high-resolution satellite images of a region in connection with change-detection applications.

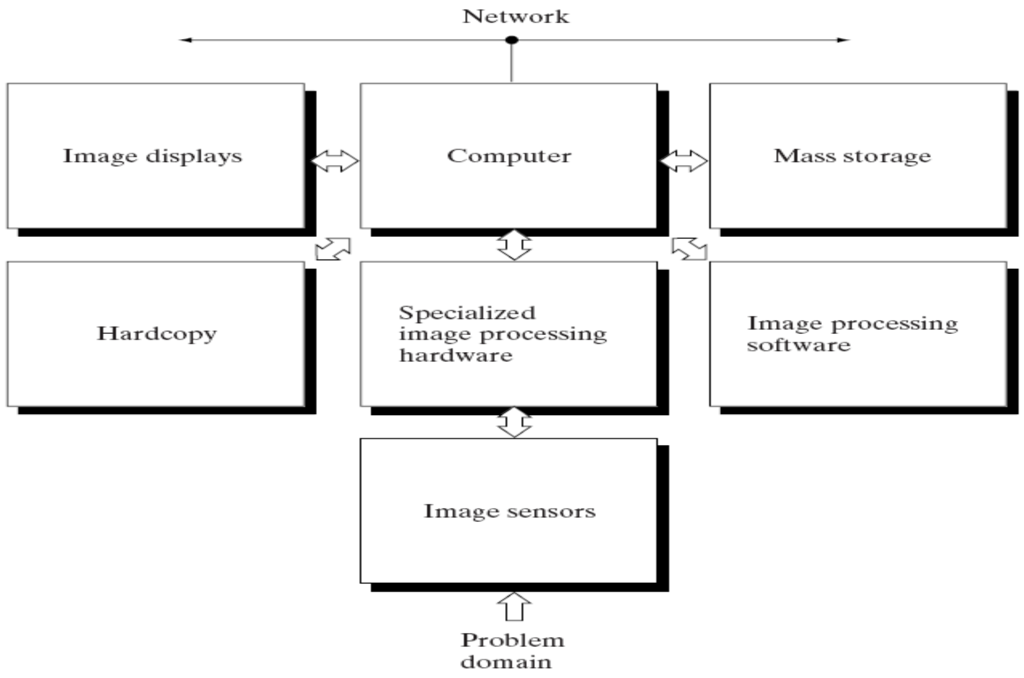
In addition to guiding the operation of each processing module, the knowledge base also controls the interaction between modules. This distinction is made in Fig. 1.23 by the use of double headed arrows between the processing modules and the knowledge base, as opposed to single-headed arrows linking the processing modules. Although we do not discuss image display explicitly at this point, it is important to keep in mind that viewing the results of image processing can take place at the output of any stage.

Components of an Image Processing System

Although large-scale image processing systems still are being sold for massive imaging applications, such as processing of satellite images, the trend continues toward miniaturizing and blending of general-purpose small computers with specialized image processing hardware.

The function of each component is discussed in the following paragraphs, starting with image sensing. With reference to *sensing*, two elements are required to acquire digital images. The first is a physical device that is sensitive to the energy radiated by the object we wish to image. The second, called a *digitizer*, is a device for converting the output of the physical sensing device into digital form. For instance, in a digital video camera, the sensors produce an electrical output proportional to light intensity. The digitizer converts these outputs to digital data.

Specialized image processing hardware usually consists of the digitizer just mentioned, plus hardware that performs other primitive operations, such as an arithmetic logic unit (ALU), which performs arithmetic and logical operations in parallel on entire images. One example of how an ALU is used is in averaging images as quickly as they are digitized, for the purpose of noise reduction. This type of hardware sometimes is called a *front-end subsystem*, and its most



distinguishing characteristic is speed. In other words, this unit performs functions that require fast data throughputs (e.g., digitizing and averaging video images at 30 frames_s) that the typical main computer cannot handle.

The ***computer*** in an image processing system is a general-purpose computer and can range from a PC to a supercomputer. In dedicated applications, sometimes specially designed computers are used to achieve a required level of performance, but our interest here is on general-purpose image processing systems. In these systems, almost any well-equipped PC-type machine is suitable for offline image processing tasks.

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

Mass storage capability is a must in image processing applications.An image of size 1024*1024 pixels, in which the intensity of each pixel is an 8-bit quantity, requires one megabyte of storage space if the image is not compressed. When dealing with thousands, or even millions, of images, providing adequate storage in an image processing system can be a challenge. Digital storage for image processing applications falls into three principal categories: (1) short term storage for use during processing, (2) on-line storage for relatively fast recall, and (3) archival storage, characterized by infrequent access. Storage is measured in bytes (eight bits), Kbytes (one thousand bytes), Mbytes (one million bytes), Gbytes (meaning giga, or one billion, bytes), and T bytes (meaning tera, or one trillion, bytes).

One method of providing short-term storage is computer memory.Another is by specialized boards, called *frame buffers,* that store one or more images and can be accessed rapidly, usually at video rates (e.g., at 30 complete images per second).The latter method allows virtually instantaneous image *zoom*, as well as *scroll* (vertical shifts) and *pan* (horizontal shifts). Frame buffers usually are housed in the specialized image processing hardware unit. Online storage generally takes the form of magnetic disks or optical-media storage. The key factor characterizing on-line storage is frequent access to the stored data. Finally, archival storage is characterized by massive storage requirements but infrequent need for access. Magnetic tapes and optical disks housed in "jukeboxes" are the usual media for archival applications.

Image displays in use today are mainly color (preferably flat screen) TV monitors. Monitors are driven by the outputs of image and graphics display cards that are an integral part of the computer system. Seldom are there requirements for image display applications that cannot be met by display cards available commercially as part of the computer system. In some cases, it is necessary to have stereo displays, and these are implemented in the form of headgear containing two small displays embedded in goggles worn by the user.

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

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

Recommended Questions

1. What is digital image processing? Explain the fundamental steps in digital image processing.
2. Briefly explain the components of an image processing system.
3. How is image formed in an eye? Explain with examples the perceived brightness is not a simple function of intensity.
4. Explain the importance of brightness adaption and discrimination in image processing.
5. Define spatial and gray level resolution. Briefly discuss the effects resulting from a reduction in number of pixels and gray levels.
6. What are the elements of visual perception?

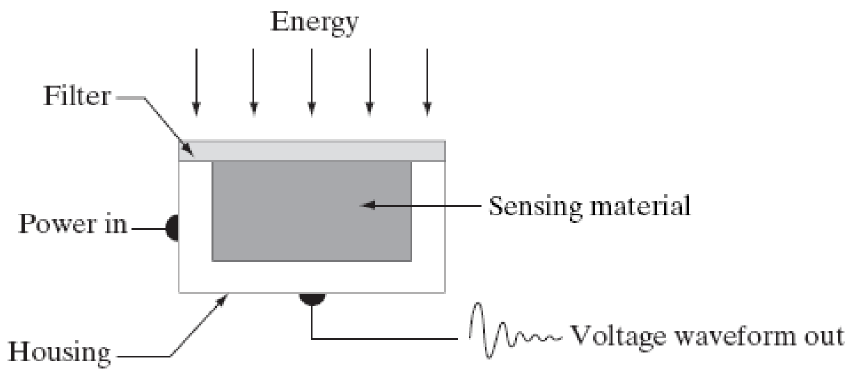
UNIT – 2

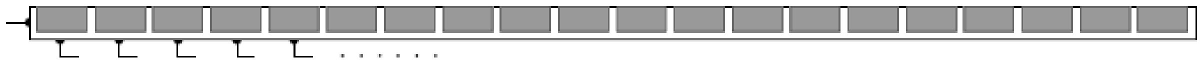
Image Sensing and Acquisition,

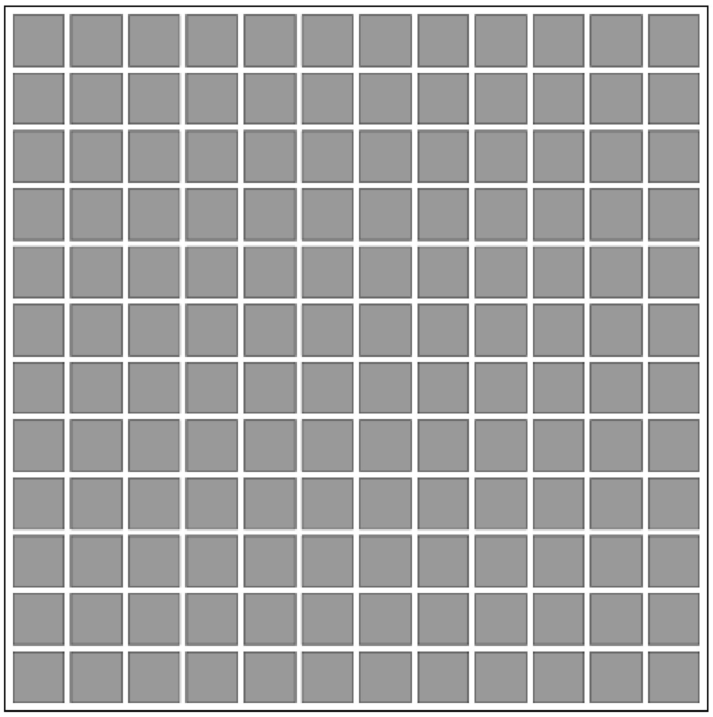
The types of images in which we are interested are generated by the combination of an “illumination” source and the reflection or absorption of energy from that source by the elements of the “scene” being imaged. We enclose *illumination* and *scene* in quotes to emphasize the fact that they are considerably more general than the familiar situation in which a visible light source illuminates a common everyday 3-D (three-dimensional) scene. For example, the illumination may originate from a source of electromagnetic energy such as radar, infrared, or X-ray energy. But, as noted earlier, it could originate from less traditional sources, such as ultrasound or even a computer-generated illumination pattern. Similarly, the scene elements could be familiar objects, but they can just as easily be molecules, buried rock formations, or a human brain. We could even image a source, such as acquiring images of the sun.

Depending on the nature of the source, illumination energy is reflected from, or transmitted through, objects. An example in the first category is light reflected from a planar surface. An example in the second category is when X-rays pass through a patient’s body for thepurpose of generating a diagnostic X-ray film. In some applications, the reflected or transmitted energy is focused onto a photo converter (e.g., a phosphor screen), which converts the energy into visible light. Electron microscopy and some applications of gamma imaging use this approach.
The idea is simple: Incoming energy is transformed into a voltage by the combination of input electrical power and sensor material that is responsive to the particular type of energy being detected.

The output voltage waveform is the response of the sensor(s), and a digital quantity is obtained from each sensor by digitizing its response. In this section, we look at the principal modalities for image sensing and generation.







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

Image Acquisition Using a Single Sensor

The components of a single sensor. Perhaps the most familiar sensor of this type is the photodiode, which is constructed of silicon materials and whose output voltage waveform is proportional to light. The use of a filter in front of a sensor improves selectivity. For example, a green (pass) filter in front of a light sensor favors light in the green band of the color spectrum. As a consequence, the sensor output will be stronger for green light than for other components in the visible spectrum. In order to generate a 2-D image using a single sensor, there has to be relative displacements in both the x- and y-directions between the sensor and the area to be imaged. Figure 2.13 shows an arrangement used in high-precision scanning, where a film negative is mounted onto a drum whose mechanical rotation provides displacement in one dimension. The single sensor is mounted on a lead screw that provides motion in the perpendicular direction. Since mechanical motion can be controlled with high precision, this method is an inexpensive (but slow) way to obtain high-resolution images. Other similar mechanical arrangements use a flat bed, with the sensor moving in two linear directions. These types of mechanical digitizers sometimes are referred to as *microdensitometers*.

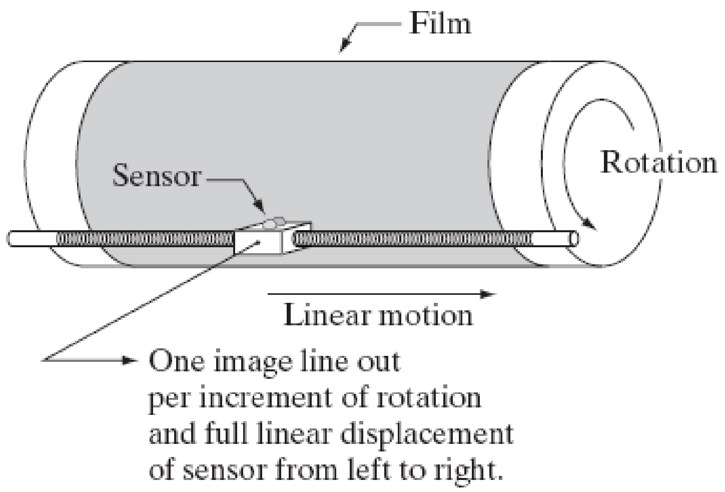
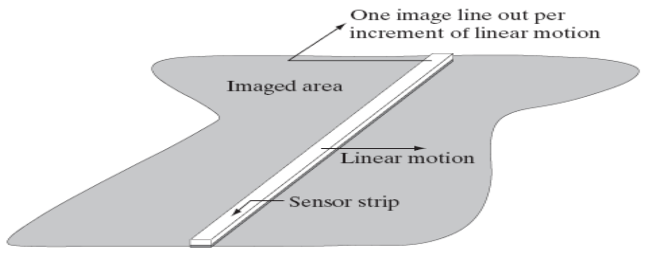

Film
Sensor
Rotation
Linear motion
One image line out per increment of rotation and full linear displacement of sensor from left to right.

Image Acquisition Using Sensor Strips

A geometry that is used much more frequently than single sensors consists of an in-line arrangement of sensors in the form of a sensor strip, shows. The strip provides imaging elements in one direction. Motion perpendicular to the strip provides imaging in the other direction. This is the type of arrangement used in most flat bed scanners. Sensing devices with 4000 or more in-line sensors are possible. In-line sensors are used routinely in airborne imaging applications, in which the imaging system is mounted on an aircraft that flies at a constant altitude and speed over the geographical area to be imaged. One-dimensional imaging sensor strips that respond to various bands of the electromagnetic spectrum are mounted perpendicular to the direction of flight. The imaging strip gives one line of an image at a time, and the motion of the strip completes the other dimension of a two-dimensional image. Lenses or other focusing schemes are used to project area to be scanned onto the sensors.

Sensor strips mounted in a ring configuration are used in medical and industrial imaging to obtain cross-sectional (“slice”) images of 3-D objects\



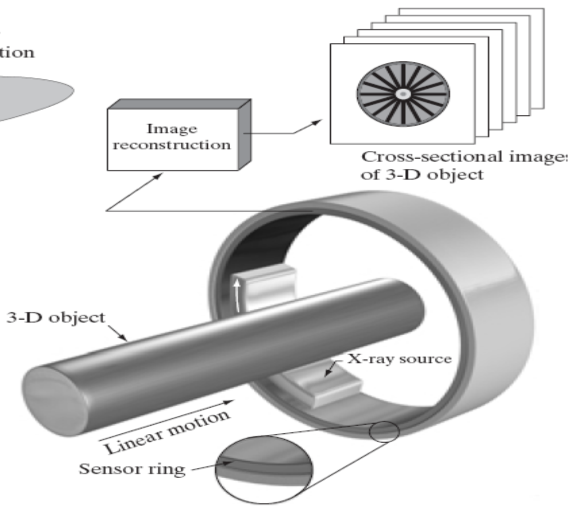


Image Acquisition Using Sensor Arrays

The individual sensors arranged in the form of a 2-D array. Numerous electromagnetic and some ultrasonic sensing devices frequently are arranged in an array format. This is also the predominant arrangement found in digital cameras. A typical sensor for these cameras is a CCD array, which can be manufactured with a broad range of sensing properties and can be packaged in rugged arrays of elements or more. CCD sensors are used widely in digital cameras and other light sensing instruments. The response of each sensor is proportional to the integral of the light energy projected onto the surface of the sensor, a property that is used in astronomical and other applications requiring low noise images. Noise reduction is achieved by letting the sensor integrate the input light signal over minutes or even hours. The two dimensional, its key advantage is that a complete image can be obtained by focusing the energy pattern onto the surface of the array. Motion obviously is not necessary, as is the case with the sensor arrangements

This figure shows the energy from an illumination source being reflected from a scene element, but, as mentioned at the beginning of this section, the energy also could be transmitted through the scene elements. The first function performed by the imaging system is to collect the incoming energy and focus it onto an image plane. If the illumination is light, the front end of the imaging system is a lens, which projects the viewed scene onto the lens focal plane. The sensor array, which is coincident with the focal plane, produces outputs proportional to the integral of the light received at each sensor. Digital and analog circuitry sweep these outputs and convert them to a video signal, which is then digitized by another section of the imaging system.

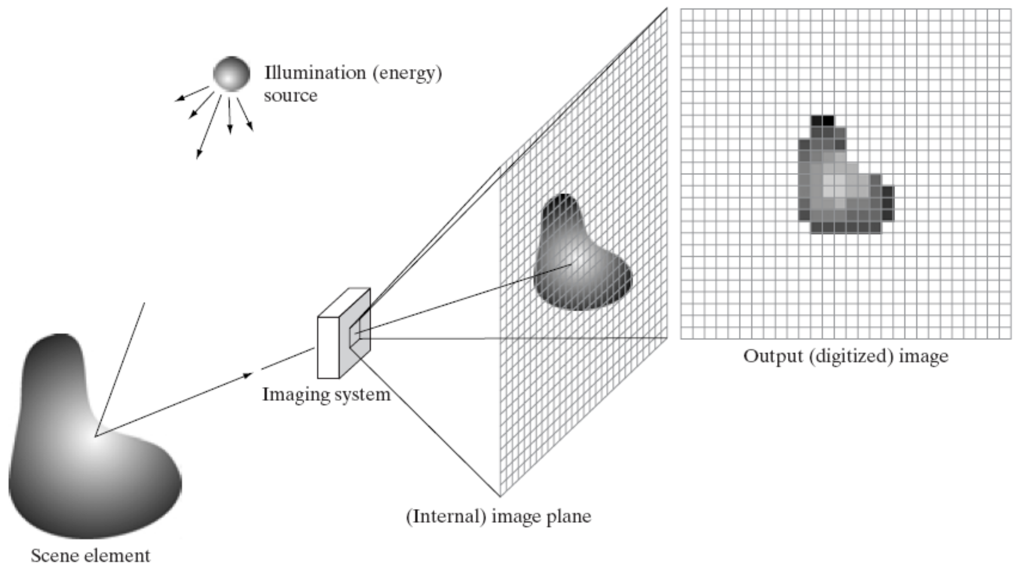
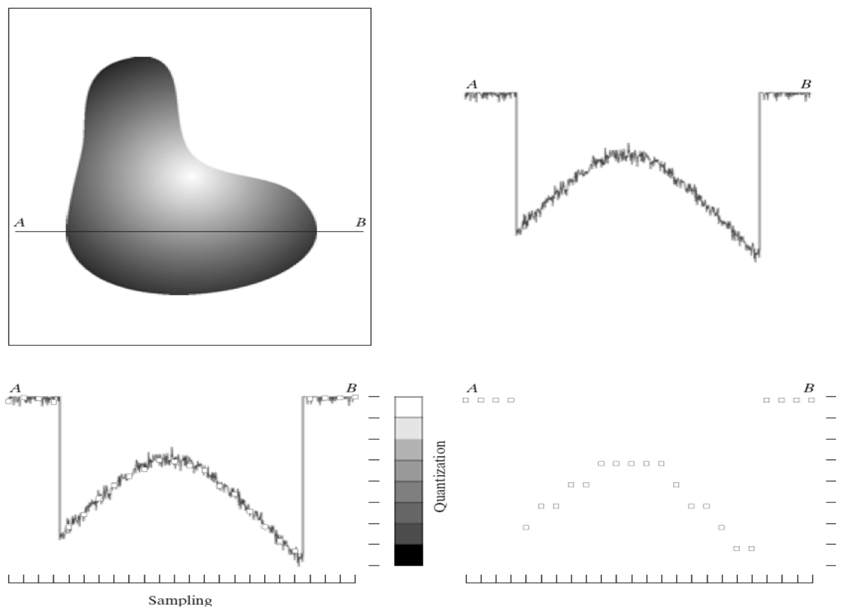


Image Sampling and Quantization,

To create a digital image, we need to convert the continuous sensed data into digital form. This involves two processes: *sampling* and *quantization*. A continuous image, f(x, y), that we want to convert to digital form. An image may be continuous with respect to the x- and y-coordinates, and also in amplitude. To convert it to digital form, we have to sample the function in both coordinates and in amplitude. Digitizing the coordinate values is called *sampling*. Digitizing the amplitude values is called *quantization*.



The one-dimensional function shown in Fig. 2.16(b) is a plot of amplitude (gray level) values of the continuous image along the line segment AB. The random variations are due to image noise. To sample this function, we take equally spaced samples along line AB, The location of each sample is given by a vertical tick mark in the bottom part of the figure. The samples are shown as small white squares superimposed on the function. The set of these discrete locations gives the sampled function. However, the values of the samples still span (vertically) a continuous range of gray-level values. In order to form a digital function, the gray-level values also must be converted (*quantized*) into discrete quantities. The right side gray-level scale divided into eight discrete levels, ranging from black to white. The vertical tick marks indicate the specific value assigned to each of the eight gray levels. The continuous gray levels are quantized simply by assigning one of the eight discrete gray levels to each sample. The assignment is made depending on the vertical proximity of a sample to a vertical tick mark. The digital samples resulting from both sampling and quantization.

Some Basic Relationships Between Pixels

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

Neighbors of a Pixel

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

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

This set of pixels, called the 4-*neighbors* of p, is denoted by $N_4(p)$. Each pixel is a unit distance from (x, y), and some of the neighbors of p lie outside the digital image if (x, y) is on the border of the image.

The four *diagonal* neighbors of p have coordinates

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

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

Adjacency, Connectivity, Regions, and Boundaries

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

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

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

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

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

(i) q is in *N*4(p), *or*

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

Linear and Nonlinear Operations

Let *H* be an operator whose input and output are images. *H* is said to be a *linear* operator if, for any two images f and g and any two scalars a and b,
H(af + bg) = aH(f) + bH(g).

In other words, the result of applying a linear operator to the sum of two images (that have been multiplied by the constants shown) is identical to applying the operator to the images individually, multiplying the results by the appropriate constants, and then adding those results. For example, an operator whose function is to compute the sum of K images is a linear operator. An operator that computes the absolute value of the difference of two images is not.

Linear operations are exceptionally important in image processing because they are based on a significant body of well-understood theoretical and practical results. Although nonlinear operations sometimes offer better performance, they are not always predictable, and for the most part are not well understood theoretically.

Recommended Questions

1. Explain the concept of sampling and quantization of an image.
2. Explain i) false contouring ii) checkboard pattern
3. How image is acquired using a single sensor? Discuss.
4. Explain zooming and shrinking digital images.
5. Define 4-adjacency, 8 – adjacency and m – adjacency.
6. With a suitable diagram, explain how an image is acquired using a circular sensor strip.
7. Explain the relationships between pixels . and also the image operations on a pixel basis.
8. Explain linear and nonlinear operations.

Unit-3

UNITARY TRANSFORMS

One dimensional signals

For a one dimensional sequence $\{f(x), 0 \le x \le N-1\}$ represented as a vector $\underline{f} = \left[f(0)\ f(1) \ldots f(N-1)\right]^T$ of size N, a transformation may be written as

$$\underline{g} = \underline{T} \cdot \underline{f} \Rightarrow g(u) = \sum_{x=0}^{N-1} T(u,x) f(x), 0 \le u \le N-1$$

where $g(u)$ is the transform (or transformation) of $f(x)$, and $T(u,x)$ is the so called

Forward transformation kernel. Similarly, the inverse transform is the relation

$$f(x) = \sum_{u=0}^{N-1} I(x,u) g(u),\ 0 \le x \le N-1$$

or written in a matrix form

$$\underline{f} = \underline{I} \cdot \underline{g} = \underline{T}^{-1} \cdot \underline{g}$$

where $I(x,u)$ is the so called **inverse transformation kernel**.

If

$$\underline{I} = \underline{T}^{-1} = \underline{T}^{*T}$$

the matrix $\underline{T}$ is called **unitary**, and the transformation is called unitary as well. It can be proven (**how?**) that the columns (or rows) of an $N \times N$ unitary matrix are orthonormal and therefore, form a complete set of **basis vectors** in the $N-$dimensional vector space.
In that case

$$\underline{f} = \underline{T}^{*T} \cdot \underline{g} \Rightarrow f(x) = \sum_{u=0}^{N-1} T^*(u,x) g(u)$$

The columns of $\underline{T}^{*T}$, that is, the vectors $\underline{T}_u^* = \left[T^*(u,0)\, T^*(u,1) \ldots T^*(u,N-1)\right]^T$ are called the **basis vectors** of $\underline{T}$.

Two dimensional signals (images)

As a one dimensional signal can be represented by an orthonormal set of **basis vectors**, an image can also be expanded in terms of a discrete set of **basis arrays** called basis images through a **two dimensional (image) transform**.

For an $N \times N$ image $f(x,y)$ the forward and inverse transforms are given below

$$g(u,v) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} T(u,v,x,y) f(x,y)$$
$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} I(x,y,u,v) g(u,v)$$

where, again, $T(u,v,x,y)$ and $I(x,y,u,v)$ are called the **forward and inverse transformation kernels**, respectively.

The forward kernel is said to be **separable** if

$$T(u,v,x,y) = T_1(u,x)T_2(v,y)$$

It is said to be **symmetric** if T_1 is functionally equal to T_2 such that

$$T(u,v,x,y) = T_1(u,x)T_1(v,y)$$

The same comments are valid for the inverse kernel.
If the kernel $T(u,v,x,y)$ of an image transform is separable and symmetric, then the transform $g(u,v) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} T(u,v,x,y) f(x,y) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} T_1(u,x)T_1(v,y) f(x,y)$ can be written in matrix form as follows

$$\underline{g} = \underline{T}_1 \cdot \underline{f} \cdot \underline{T}_1^{\,T}$$

where $\underline{f}$ is the original image of size $N \times N$, and $\underline{T}_1$ is an $N \times N$ transformation matrix with elements $t_{ij} = T_1(i,j)$. If, in addition, $\underline{T}_1$ is a unitary matrix then the transform is called **separable unitary** and the original image is recovered through the relationship

$$\underline{f} = \underline{T}_1^{\,*T} \cdot \underline{g} \cdot \underline{T}_1^{\,*}$$

Fundamental properties of unitary transforms

The property of energy preservation

In the unitary transformation

$$\underline{g} = \underline{T} \cdot \underline{f}$$

it is easily proven (try the proof by using the relation $\underline{T}^{-1} = \underline{T}^{*T}$) that

$$\|\underline{g}\|^2 = \|\underline{f}\|^2$$

Thus, a unitary transformation preserves the signal energy. This property is called energy preservation property.

This means that every unitary transformation is simply a rotation of the vector $\underline{f}$ in the N - dimensional vector space.

For the 2-D case the energy preservation property is written as

$$\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}\left|f(x,y)\right|^2=\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\left|g(u,v)\right|^2$$

The property of energy compaction

Most unitary transforms pack a large fraction of the energy of the image into relatively few of the transform coefficients. This means that relatively few of the transform coefficients have significant values and these are the coefficients that are close to the origin (small index coefficients).

This property is very useful for compression purposes. (**Why?**)

THE TWO DIMENSIONAL FOURIER TRANSFORM

Continuous space and continuous frequency

The Fourier transform is extended to a function $f(x,y)$ of two variables. If $f(x,y)$ is continuous and integrable and $F(u,v)$ is integrable, the following Fourier transform pair exists:

$$F(u,v)=\int_{-\infty}^{\infty}\int f(x,y)e^{-j2\pi(ux+vy)}dxdy$$

$$f(x,y)=\frac{1}{(2\pi)^2}\int_{-\infty}^{\infty}\int F(u,v)e^{j2\pi(ux+vy)}dudv$$

In general $F(u,v)$ is a complex-valued function of two real frequency variables u,v and hence, it can be written as:

$$F(u,v)=R(u,v)+jI(u,v)$$

The amplitude spectrum, phase spectrum and power spectrum, respectively, are defined as follows.

$$\left|F(u,v)\right|=\sqrt{R^2(u,v)+I^2(u,v)}$$

$$\phi(u,v)=\tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$$

$$P(u,v)=\left|F(u,v)\right|^2=R^2(u,v)+I^2(u,v)$$

2.2 Discrete space and continuous frequency

For the case of a discrete sequence $f(x,y)$ of infinite duration we can define the 2-D discrete space Fourier transform pair as follows

$$F(u,v)=\sum_{x=-\infty}^{\infty}\sum_{y=-\infty}^{\infty}f(x,y)e^{-j(xu+vy)}$$

$$f(x,y)=\frac{1}{(2\pi)^2}\int_{u=-\pi}^{\pi}\int_{v=-\pi}^{\pi}F(u,v)e^{j(xu+vy)}dudv$$

$F(u,v)$ is again a complex-valued function of two real frequency variables u,v and it is

periodic with a period $2\pi\times 2\pi$, that is to say

$$F(u,v)=F(u+2\pi,v)=F(u,v+2\pi)$$

The Fourier transform of $f(x,y)$ is said to converge uniformly when $F(u,v)$ is finite and

$$\lim_{N_1\to\infty}\lim_{N_2\to\infty}\sum_{x=-N_1}^{N_1}\sum_{y=-N_2}^{N_2}f(x,y)e^{-j(xu+vy)}=F(u,v)\text{ for all }u,v.$$

When the Fourier transform of $f(x,y)$ converges uniformly, $F(u,v)$ is an analytic function and is infinitely differentiable with respect to u and v.

Discrete space and discrete frequency: The two dimensional Discrete Fourier Transform (2-D DFT)

If $f(x,y)$ is an $M\times N$ array, such as that obtained by sampling a continuous function of two dimensions at dimensions M and N on a rectangular grid, then its two dimensional Discrete Fourier transform (DFT) is the array given by

$$F(u,v)=\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)e^{-j2\pi(ux/M+vy/N)}$$

$$u=0,\ldots,M-1,\ v=0,\ldots,N-1$$

and the inverse DFT (IDFT) is

$$f(x,y)=\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v)e^{j2\pi(ux/M+vy/N)}$$

When images are sampled in a square array, $M=N$ and

$$F(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)e^{-j2\pi(ux+vy)/N}$$

$$f(x,y)=\frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}F(u,v)e^{j2\pi(ux+vy)/N}$$

It is straightforward to prove that the two dimensional Discrete Fourier Transform is separable, symmetric and unitary.

Properties of the 2-D DFT

Most of them are straightforward extensions of the properties of the 1-D Fourier Transform. Advise any introductory book on Image Processing.

The importance of the phase in 2-D DFT. Image reconstruction from amplitude or phase only.

The Fourier transform of a sequence is, in general, complex-valued, and the unique representation of a sequence in the Fourier transform domain requires both the phase and the magnitude of the Fourier transform. In various contexts it is often desirable to reconstruct a signal from only partial domain information. Consider a 2-D sequence $f(x,y)$ with Fourier transform $F(u,v)=\Im\{f(x,y)\}$ so that

$$F(u,v)=\Im\{f(x,y\}=|F(u,v)|e^{j\phi_f(u,v)}$$

It has been observed that a straightforward signal synthesis from the Fourier transform phase $\phi_f(u,v)$ alone, often captures most of the intelligibility of the original image $f(x,y)$ (**why?**). A straightforward synthesis from the Fourier transform magnitude $|F(u,v)|$ alone, however, does not generally capture the original signal's intelligibility. The above observation is valid for a large number of signals (or images). To illustrate this, we can synthesise the phase-only signal $f_p(x,y)$ and the magnitude-only signal $f_m(x,y)$ by

$$f_p(x,y)=\Im^{-1}\left[e^{j\phi_f(u,v)}\right]$$

$$f_m(x,y)=\Im^{-1}\left[|F(u,v)|e^{j0}\right]$$

and observe the two results (**Try this exercise in MATLAB**).

An experiment which more dramatically illustrates the observation that phase-only signal synthesis captures more of the signal intelligibility than magnitude-only synthesis, can be performed as follows.

Consider two images $f(x,y)$ and $g(x,y)$. From these two images, we synthesise two other images $f_1(x,y)$ and $g_1(x,y)$ by mixing the amplitudes and phases of the original images as follows:

$$f_1(x,y) = \Im^{-1}\left\{|G(u,v)|e^{j\phi_f(u,v)}\right\}$$
$$g_1(x,y) = \Im^{-1}\left\{|F(u,v)|e^{j\phi_g(u,v)}\right\}$$

In this experiment $f_1(x,y)$ captures the intelligibility of $f(x,y)$, while $g_1(x,y)$ captures the intelligibility of $g(x,y)$ (**Try this exercise in MATLAB**).

<u>THE DISCRETE COSINE TRANSFORM (DCT)</u>

One dimensional signals

This is a transform that is similar to the Fourier transform in the sense that the new independent variable represents again frequency. The DCT is defined below.

$$C(u) = a(u)\sum_{x=0}^{N-1} f(x)\cos\left[\frac{(2x+1)u\pi}{2N}\right],\ u = 0,1,\ldots,N-1$$

with $a(u)$ a parameter that is defined below.

$$a(u) = \begin{cases} \sqrt{1/N} & u = 0 \\ \\ \sqrt{2/N} & u = 1,\ldots,N-1 \end{cases}$$

The inverse DCT (IDCT) is defined below.

$$f(x) = \sum_{u=0}^{N-1} a(u)C(u)\cos\left[\frac{(2x+1)u\pi}{2N}\right]$$

Two dimensional signals (images)

For 2-D signals it is defined as

$$C(u,v) = a(u)a(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$
$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} a(u)a(v)C(u,v)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$a(u)$ is defined as above and $u,v = 0,1,\ldots,N-1$

Properties of the DCT transform

- The DCT is a real transform. This property makes it attractive in comparison to the Fourier transform.
- The DCT has excellent energy compaction properties. For that reason it is widely used in image compression standards (as for example JPEG standards).
- There are fast algorithms to compute the DCT, similar to the FFT for computing the DFT.

Recommended Questions

1. Define two-dimensional DFT. Explain the following properties of 2-DFT.

i) Translation ii) rotation iii) distributivity and scaling iv) separability

2. What are basis vectors?
3. Derive the expression for 2D circular convolution theorem.
4. Define two – dimensional unitary transform. Check whether the unitary DFT matrix is unitary or not for N = 4.
5. For the 2 X 2 transform A and the image U

$$A = 1/1 \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad \text{and } U = \begin{bmatrix} 1 & 2 \\ 8 & 4 \end{bmatrix}$$

Calculate the transformed image V and the basis images.

6. Consider the image segment shown in fig

i) Let V = {0, 1}. Compute the lengths of shortest 4 - , 8 – and m – paths between p and q.

ii) Repeat for V = {1, 2}.

$$\begin{array}{c} \\ \\ (p) \end{array}\begin{bmatrix} 3 & 1 & 2 & 1 \\ 3 & 2 & 0 & 2 \\ 1 & 2 & 1 & 1 \end{bmatrix}\begin{array}{c} (q) \\ \\ \\ \end{array}$$

UNIT – 4

WALSH TRANSFORM (WT)

One dimensional signals

This transform is slightly different from the transforms you have met so far. Suppose we have a function $f(x), x = 0,\ldots, N-1$ where $N = 2^n$ and its Walsh transform $W(u)$.

If we use binary representation for the values of the independent variables x and u we

need n bits to represent them. Hence, for the binary representation of x and u we can

write: $(x)_{10} = (b_{n-1}(x)b_{n-2}(x)\ldots b_0(x))_2$, $(u)_{10} = (b_{n-1}(u)b_{n-2}(u)\ldots b_0(u))_2$ with $b_i(x)\ 0 \text{ or } 1$ for $i = 0,\ldots, n-1$.

Example

If $f(x), x = 0,\ldots,7, (8\,\text{samples})$ then $n = 3$ and for $x = 6$, $6 = (110)_2 \Rightarrow b_2(6) = 1, b_1(6) = 1, b_0(6) = 0$

We define now the 1-D Walsh transform as

$$W(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}\right] \text{ or}$$

$$W(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)(-1)^{\sum_{i=0}^{n-1} b_i(x)b_{n-1-i}(u)}$$

The array formed by the Walsh kernels is again a symmetric matrix having orthogonal rows and columns. Therefore, the Walsh transform is and its elements are of the form $T(u,x) = \prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$. You can immediately observe that $T(u,x) = 0$ or 1 depending on the values of $b_i(x)$ and $b_{n-1-i}(u)$. If the Walsh transform is written in a matrix form

$$\underline{W} = \underline{T} \cdot \underline{f}$$

for $u = 0$ we see that $(u)_{10} = (b_{n-1}(u)b_{n-2}(u)\ldots b_0(u))_2 = (00\ldots 0)_2$ and hence, $b_{n-1-i}(u) = 0$, for any i. Thus, $T(0,x) = 1$ and $W(0) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)$. We see that the first element of the Walsh transform in the mean of the original function $f(x)$ (the DC value) as it is the case with the Fourier transform.

The inverse Walsh transform is defined as follows.

$$f(x) = \sum_{u=0}^{N-1} W(u) \left[\prod_{i=0}^{n-1} (-1)^{b_i(x) b_{n-1-i}(u)} \right] \text{ or}$$

$$f(x) = \sum_{u=0}^{N-1} W(u) (-1)^{\sum_{i=0}^{n-1} b_i(x) b_{n-1-i}(u)}$$

Two dimensional signals

The Walsh transform is defined as follows for two dimensional signals.

$$W(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \left[\prod_{i=0}^{n-1} (-1)^{(b_i(x) b_{n-1-i}(u) + b_i(y) b_{n-1-i}(v))} \right] \text{ or}$$

$$W(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) (-1)^{\sum_{i=0}^{n-1} (b_i(x) b_{n-1-i}(u) + b_i(y) b_{n-1-i}(v))}$$

The inverse Walsh transform is defined as follows for two dimensional signals.

$$f(x,y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} W(u,v) \left[\prod_{i=0}^{n-1} (-1)^{(b_i(x) b_{n-1-i}(u) + b_i(y) b_{n-1-i}(v))} \right] \text{ or}$$

$$f(x,y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} W(u,v) (-1)^{\sum_{i=0}^{n-1} (b_i(x) b_{n-1-i}(u) + b_i(y) b_{n-1-i}(v))}$$

4.3 Properties of the Walsh Transform

- Unlike the Fourier transform, which is based on trigonometric terms, the Walsh transform consists of a series expansion of basis functions whose values are only -1 or 1 and they have the form of square waves. These functions can be implemented more efficiently in a digital environment than the exponential basis functions of the Fourier transform.
- The forward and inverse Walsh kernels are identical except for a constant multiplicative factor of $\frac{1}{N}$ for 1-D signals.
- The forward and inverse Walsh kernels are identical for 2-D signals. This is because the array formed by the kernels is a symmetric matrix having orthogonal rows and columns, so its inverse array is the same as the array itself.

- The concept of frequency exists also in Walsh transform basis functions. We can think of frequency as the number of zero crossings or the number of transitions in a basis vector and we call this number **sequency**. The Walsh transform exhibits the property of energy compaction as all the transforms that we are currently studying. (**why**?)
- For the fast computation of the Walsh transform there exists an algorithm called **Fast Walsh Transform (FWT).** This is a straightforward modification of the FFT. Advise any introductory book for your own interest.

HADAMARD TRANSFORM (HT)

Definition

In a similar form as the Walsh transform, the 2-D Hadamard transform is defined as follows.

Forward

$$H(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_i(u)+b_i(y)b_i(v))}\right],\ N=2^n \text{ or}$$

$$H(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)(-1)^{\sum_{i=0}^{n-1}(b_i(x)b_i(u)+b_i(y)b_i(v))}$$

Inverse

$$f(x,y)=\frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}H(u,v)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_i(u)+b_i(y)b_i(v))}\right] \text{ etc.}$$

Properties of the Hadamard Transform

- **Most of the comments made for Walsh transform are valid here.**
- The Hadamard transform differs from the Walsh transform only in the order of basis functions. The order of basis functions of the Hadamard transform **does not** allow the fast computation of it by using a straightforward modification of the FFT. An extended version of the Hadamard transform is the **Ordered Hadamard Transform** for which a fast algorithm called **Fast Hadamard Transform (FHT)** can be applied.

- An important property of Hadamard transform is that, letting H_N represent the matrix of order N, the recursive relationship is given by the expression

$$H_{2N} = \begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix}$$

KARHUNEN-LOEVE (KLT) or HOTELLING TRANSFORM

The Karhunen-Loeve Transform or KLT was originally introduced as a series expansion for continuous random processes by Karhunen and Loeve. For discrete signals Hotelling first studied what was called a method of principal components, which is the discrete equivalent of the KL series expansion. Consequently, the KL transform is also called the Hotelling transform or the method of principal components. **The term KLT is the most widely used**.

The case of many realisations of a signal or image (Gonzalez/Woods)

The concepts of **eigenvalue** and **eigevector** are necessary to understand the KL transform.

If $\underline{C}$ is a matrix of dimension $n \times n$, then a scalar λ is called an eigenvalue of $\underline{C}$ if there is a nonzero vector $\underline{e}$ in R^n such that

$$\underline{C}\underline{e} = \lambda \underline{e}$$

The vector $\underline{e}$ is called an eigenvector of the matrix $\underline{C}$ corresponding to the eigenvalue λ.

(If you have difficulties with the above concepts consult any elementary linear algebra book.)

Consider a population of random vectors of the form

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

The **mean vector** of the population is defined as

$$\underline{m}_x = E\{\underline{x}\}$$

The operator E refers to the expected value of the population calculated theoretically using the probability density functions (pdf) of the elements x_i and the joint probability density functions between the elements x_i and x_j.

The covariance matrix of the population is defined as

$$\underline{C}_x = E\{(\underline{x} - \underline{m}_x)(\underline{x} - \underline{m}_x)^T\}$$

Because $\underline{x}$ is n-dimensional, $\underline{C}_x$ and $(\underline{x}-\underline{m}_x)(\underline{x}-\underline{m}_x)^T$ are matrices of order $n \times n$. The element c_{ii} of $\underline{C}_x$ is the variance of x_i, and the element c_{ij} of $\underline{C}_x$ is the covariance between the elements x_i and x_j. If the elements x_i and x_j are uncorrelated, their covariance is zero and, therefore, $c_{ij} = c_{ji} = 0$.

For M vectors from a random population, where M is large enough, the mean vector and covariance matrix can be approximately calculated from the vectors by using the following relationships where all the expected values are approximated by summations

$$\underline{m}_x = \frac{1}{M}\sum_{k=1}^{M}\underline{x}_k$$

$$\underline{C}_x = \frac{1}{M}\sum_{k=1}^{M}\underline{x}_k\underline{x}_k^T - \underline{m}_x\underline{m}_x^T$$

Very easily it can be seen that $\underline{C}_x$ is real and symmetric. In that case a set of n orthonormal (**at this point you are familiar with that term**) eigenvectors always exists. Let $\underline{e}_i$ and λ_i, $i = 1,2,\ldots,n$, be this set of eigenvectors and corresponding eigenvalues of $\underline{C}_x$, arranged in descending order so that $\lambda_i \geq \lambda_{i+1}$ for $i = 1,2,\ldots,n-1$. Let $\underline{A}$ be a matrix whose rows are formed from the eigenvectors of $\underline{C}_x$, ordered so that the first row of $\underline{A}$ is the eigenvector corresponding to the largest eigenvalue, and the last row the eigenvector corresponding to the smallest eigenvalue.

Suppose that $\underline{A}$ is a transformation matrix that maps the vectors $\underline{x}'s$ into vectors $\underline{y}'s$ by using the following transformation

$$\underline{y} = \underline{A}(\underline{x} - \underline{m}_x)$$

The above transform is called the **Karhunen-Loeve** or **Hotelling** transform. The mean of the $\underline{y}$ vectors resulting from the above transformation is zero (**try to prove that**)

$$\underline{m}_y = \underline{0}$$

the covariance matrix is (**try to prove that**)

$$\underline{C}_y = \underline{A}\underline{C}_x\underline{A}^T$$

and $\underline{C}_y$ is a diagonal matrix whose elements along the main diagonal are the eigenvalues of $\underline{C}_x$ (**try to prove that**)

$$\underline{C}_y = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}$$

The off-diagonal elements of the covariance matrix are 0, so the elements of the $\underline{y}$ vectors are uncorrelated.

Lets try to reconstruct any of the original vectors $\underline{x}$ from its corresponding $\underline{y}$. Because the rows of $\underline{A}$ are orthonormal vectors (**why?**), then $\underline{A}^{-1} = \underline{A}^T$, and any vector $\underline{x}$ can by recovered from its corresponding vector $\underline{y}$ by using the relation

$$\underline{x} = \underline{A}^T \underline{y} + \underline{m}_x$$

Suppose that instead of using all the eigenvectors of $\underline{C}_x$ we form matrix $\underline{A}_K$ from the K eigenvectors corresponding to the K largest eigenvalues, yielding a transformation matrix of order $K \times n$. The $\underline{y}$ vectors would then be K dimensional, and the reconstruction of any of the original vectors would be approximated by the following relationship

$$\hat{\underline{x}} = \underline{A}_K^T \underline{y} + \underline{m}_x$$

The mean square error between the perfect reconstruction $\underline{x}$ and the approximate reconstruction $\hat{\underline{x}}$ is given by the expression

$$e_{ms} = \sum_{j=1}^{n} \lambda_j - \sum_{j=1}^{K} \lambda_j = \sum_{j=K+1}^{n} \lambda_j \,.$$

By using $\underline{A}_K$ instead of $\underline{A}$ for the KL transform we achieve compression of the available data.

The case of <u>one</u> realisation of a signal or image

The derivation of the KLT for the case of one image realisation assumes that the two dimensional signal (image) is **ergodic**. This assumption allows us to calculate the statistics of the image using only one realisation. Usually we divide the image into blocks and we apply the KLT in each block. This is reasonable because the 2-D field is likely to be ergodic within a small block since the nature of the signal changes within the whole image. Let’s suppose that $\underline{f}$ is a vector obtained by lexicographic ordering of the pixels $f(x, y)$ within a block of size $M \times M$ (placing the rows of the block sequentially).

The mean vector of the random field inside the block is a scalar that is estimated by the approximate relationship

$$m_f = \frac{1}{M^2}\sum_{k=1}^{M^2} \underline{f}(k)$$

and the covariance matrix of the 2-D random field inside the block is $\underline{C}_f$ where

$$c_{ii} = \frac{1}{M^2}\sum_{k=1}^{M^2} \underline{f}(k)\underline{f}(k) - m_f^2$$

and

$$c_{ij} = c_{i-j} = \frac{1}{M^2}\sum_{k=1}^{M^2} \underline{f}(k)\underline{f}(k+i-j) - m_f^2$$

After knowing how to calculate the matrix $\underline{C}_f$, the KLT for the case of a single realisation is the same as described above.

6.3 Properties of the Karhunen-Loeve transform

Despite its favourable theoretical properties, the KLT is not used in practice for the following reasons.

- Its basis functions depend on the covariance matrix of the image, and hence they have to recomputed and transmitted for every image.
- Perfect decorrelation is not possible, since images can rarely be modelled as realisations of ergodic fields.
- There are no fast computational algorithms for its implementation.

Recommended Questions

1. Construct Haar transform matrix for N = 2.
2. Explain the importance of discrete cosine transform, with its properties.
3. Define DCT and its inverse transformation .
4. Discuss any three properties of discrete cosine transform.
5. Develop Hadamard transform for n = 3.
6. Discuss the properties of the Hadamard transform .
7. Derive the relation between DCT and DFT.
8. Write H matrix for the Haar transform for N = 8 and explain how it is constructed.

UNIT – 5

Preliminaries

Spatial domain methods

Suppose we have a digital image which can be represented by a two dimensional random field $f(x, y)$.

An image processing operator in the spatial domain may be expressed as a mathematical function $T[\,]$ applied to the image $f(x, y)$ to produce a new image $g(x, y) = T[f(x, y)]$ as follows.

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

The operator T applied on $f(x, y)$ may be defined over:

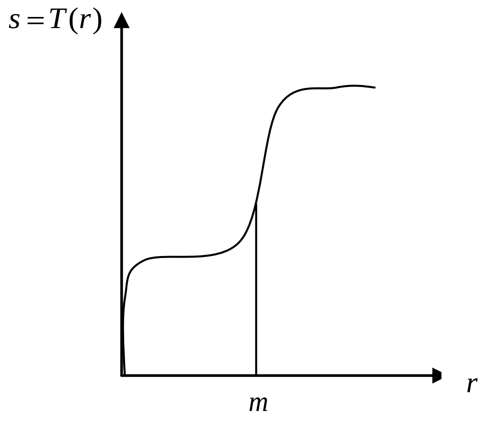
(i) A single pixel (x, y). In this case T is a grey level transformation (or mapping) function.

(ii) Some neighbourhood of (x, y).

(iii) T may operate to a set of input images instead of a single image.

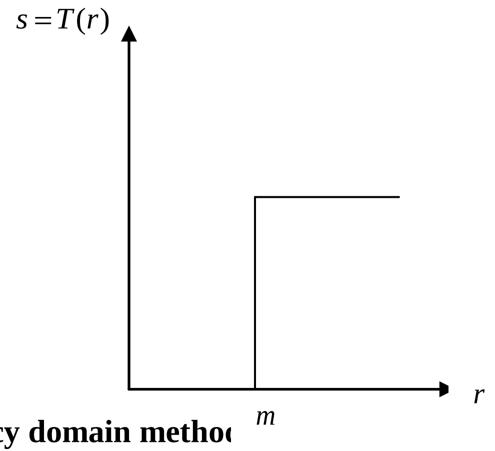
Example 1

The result of the transformation shown in the figure below is to produce an image of higher contrast than the original, by darkening the levels below m and brightening the levels above m in the original image. This technique is known as **contrast stretching**.



Example 2

The result of the transformation shown in the figure below is to produce a binary image.



Frequency domain metho

Let $g(x, y)$ be a desired image formed by the convolution of an image $f(x, y)$ and a linear, position invariant operator $h(x, y)$, that is:

$$g(x, y) = h(x, y) * f(x, y)$$

The following frequency relationship holds:

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

We can select $H(u,v)$ so that the desired image

$$g(x, y) = \Im^{-1}\{H(u,v)F(u,v)\}$$

exhibits some highlighted features of $f(x, y)$. For instance, edges in $f(x, y)$ can be accentuated by using a function $H(u,v)$ that emphasises the high frequency components of $F(u,v)$.

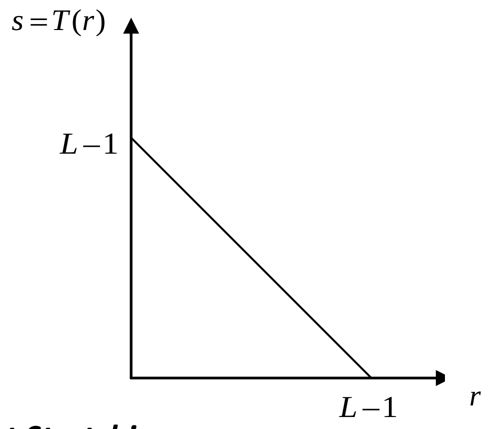
Spatial domain: Enhancement by point processing

We are dealing now with image processing methods that are based only on the intensity of single pixels.

Intensity transformations

Image Negatives

The negative of a digital image is obtained by the transformation function $s = T(r) = L - 1 - r$ shown in the following figure, where L is the number of grey levels. The idea is that the intensity of the output image decreases as the intensity of the input increases. This is useful in numerous applications such as displaying medical images.



Contrast Stretching

contrast images occur often due to poor or non uniform lighting conditions, or due to nonlinearity, or small dynamic range of the imaging sensor. In the figure of Example 1 above you have seen a typical contrast stretching transformation.

Histogram processing. Definition of the histogram of an image.

By processing (modifying) the histogram of an image we can create a new image with specific desired properties.

Suppose we have a digital image of size $N \times N$ with grey levels in the range $[0, L-1]$. The histogram of the image is defined as the following discrete function:

$$p(r_k) = \frac{n_k}{N^2}$$

Where

r_k is the kth grey level, $k = 0,1,\ldots,L-1$
n_k is the number of pixels in the image with grey level r_k
N^2 is the total number of pixels in the image

The histogram represents the frequency of occurrence of the various grey levels in the image. A plot of this function for all values of k provides a global description of the appearance of the image.

Question: Think how the histogram of a dark image, a bright image and an image of very low contrast would like. Plot its form in each case.

Global histogram equalisation

In this section we will assume that the image to be processed has a continuous intensity that lies within the interval $[0, L-1]$. Suppose we divide the image intensity with its maximum value $L-1$. Let the variable r represent the new grey levels (image intensity) in the image, where now $0 \le r \le 1$ and let $p_r(r)$ denote the probability density function (pdf) of the variable r. We now apply the following transformation function to the intensity

$$s = T(r) = \int_0^r p_r(w)dw \qquad , \qquad 0 \le r \le 1$$

(1) By observing the transformation of equation (1) we immediately see that it possesses the following properties:

(i) $0 \le s \le 1$.

(ii) $r_2 > r_1 \Rightarrow T(r_2) \ge T(r_1)$, i.e., the function $T(r)$ is increase ng with r.

(iii) $s = T(0) = \int_0^0 p_r(w)dw = 0$ and $s = T(1) = \int_0^1 p_r(w)dw = 1$. Moreover, if the original image has intensities <u>**only**</u> within a certain range $[r_{\min}, r_{\max}]$ then $s = T(r_{\min}) = \int_0^{r_{\min}} p_r(w)dw = 0$ and $s = T(r_{\max}) = \int_0^{r_{\max}} p_r(w)dw = 1$ since $p_r(r) = 0, r < r_{\min}$ and $r > r_{\max}$. Therefore, the new intensity s takes always all values within the available range [0 1].

Suppose that $P_r(r)$, $P_s(s)$ are the probability distribution functions (PDF's) of the variables r and s respectively.

Let us assume that the original intensity lies within the values r and $r + dr$ with dr a small quantity. dr can be assumed small enough so as to be able to consider the function $p_r(w)$ constant within the interval $[r, r + dr]$ and equal to $p_r(r)$. Therefore,

$$P_r[r, r+dr] = \int_r^{r+dr} p_r(w)dw \cong p_r(r) \int_r^{r+dr} dw = p_r(r)dr \ .$$

Now suppose that $s = T(r)$ and $s_1 = T(r + dr)$. The quantity dr can be assumed small enough so as to be able to consider that $s_1 = s + ds$ with ds

small enough so as to be able to consider the function $p_s(w)$ constant within the interval $[s, s+ds]$ and equal to $p_s(s)$. Therefore,

$$P_s[s, s+ds] = \int_s^{s+ds} p_s(w)dw \cong p_s(s)\int_s^{s+ds} dw = p_s(s)ds$$

Since $s = T(r)$, $s + ds = T(r+dr)$ and the function of equation (1) is increasing with r, all and only the values within the interval $[r, r+dr]$ will be mapped within the interval $[s, s+ds]$. Therefore,

$$P_r[r, r+dr] = P_s[s, s+ds] \Rightarrow p_r(r)dr \overset{r=T^{-1}(s)}{=} p_s(s)ds \Rightarrow p_s(s) = p_r(r)\left.\frac{dr}{ds}\right|_{r=T^{-1}(s)}$$

From equation (1) we see that

$$\frac{ds}{dr} = p_r(r)$$

and hence,

$$p_s(s) = \left[p_r(r)\frac{1}{p_r(r)}\right]_{r=T^{-1}(s)} = 1,\ 0 \le s \le 1$$

Conclusion

From the above analysis it is obvious that the transformation of equation (1) converts the original image into a new image with uniform probability density function. This means that in the new image all intensities are present [look at property (iii) above] and with equal probabilities. The whole range of intensities from the absolute black to the absolute white are explored and the new image will definitely have higher contrast compared to the original image.

Unfortunately, in a real life scenario we must deal with digital images. The discrete form of histogram equalisation is given by the relation

$$s_k = T(r_k) = \sum_{j=0}^{k}\frac{n_j}{N^2} = \sum_{j=0}^{k} p_r(r_j),\ 0 \le r_k \le 1,\ k = 0,1,\ldots,L-1$$

(2) The quantities in equation (2) have been defined in Section 2.2. To see results of histogram equalisation look at any introductory book on Image Processing.

The improvement over the original image is quite evident after using the technique of histogram equalisation. The new histogram is **not flat** because of the discrete approximation of the probability density function with the histogram function. Note, however, that the grey levels of an image that has been subjected to histogram equalisation are spread out and always reach white. This process increases the dynamic range of grey levels and produces an increase in image contrast.

Local histogram equalisation

Global histogram equalisation is suitable for overall enhancement. It is often necessary to enhance details over small areas. The number of pixels in these areas my have negligible influence on the computation of a global transformation, so the use of this type of transformation does not necessarily guarantee the desired local enhancement. The solution is to devise transformation functions based on the grey level distribution – or other properties – in the neighbourhood of every pixel in the image. The histogram processing technique previously described is easily adaptable to local enhancement. The procedure is to define a square or rectangular neighbourhood and move the centre of this area from pixel to pixel. At each location the histogram of the points in the neighbourhood is computed and a histogram equalisation transformation function is obtained. This function is finally used to map the grey level of the pixel centred in the neighbourhood.

The centre of the neighbourhood region is then moved to an adjacent pixel location and the procedure is repeated. Since only one new row or column of the neighbourhood changes during a pixel-to-pixel translation of the region, updating the histogram obtained in the previous location with the new data introduced at each motion step is possible quite easily. This approach has obvious advantages over repeatedly computing the histogram over all pixels in the neighbourhood region each time the region is moved one pixel location. Another approach often used to reduce computation is to utilise non overlapping regions, but this methods usually produces an undesirable checkerboard effect.

Histogram specification

Suppose we want to specify a particular histogram shape (not necessarily uniform) which is capable of highlighting certain grey levels in the image.

Let us suppose that:

$p_r(r)$ is the original probability density function

$p_z(z)$ is the desired probability density function

Suppose that histogram equalisation is first applied on the original image r

$$s = T(r) = \int_0^r p_r(w)dw$$

Suppose that the desired image z is available and histogram equalisation is applied as well

$$v = G(z) = \int_0^z p_z(w)dw$$

$p_s(s)$ and $p_v(v)$ are both uniform densities and they can be considered as identical. Note that the final result of histogram equalisation is independent of the density inside the integral. So in equation $v = G(z) = \int_0^z p_z(w)dw$ we can use the symbol s instead of v.

The inverse process $z = G^{-1}(s)$ will have the desired probability density function. Therefore, the process of histogram specification can be summarised in the following steps.

(i) We take the original image and equalise its intensity using the relation $s = T(r) = \int_0^r p_r(w)dw$.

(ii) From the given probability density function $p_z(z)$ we specify the probability distribution function $G(z)$.

(iii) We apply the inverse transformation function $z = G^{-1}(s) = G^{-1}\left[T(r)\right]$

Spatial domain: Enhancement in the case of many realisations of an image of interest available

Image averaging

Suppose that we have an image $f(x,y)$ of size $M \times N$ pixels corrupted by noise $n(x,y)$, so we obtain a noisy image as follows.

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

For the noise process $n(x,y)$ the following assumptions are made.

(i) The noise process $n(x,y)$ is ergodic.

(ii) It is zero mean, i.e., $E\{n(x,y)\} = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} n(x,y) = 0$

(ii) It is white, i.e., the autocorrelation function of the noise process defined as $R[k,l] = E\{n(x,y)n(x+k,y+l)\} = \frac{1}{(M-k)(N-l)}\sum_{x=0}^{M-1-k}\sum_{y=0}^{N-1-l} n(x,y)n(x+k,y+l)$ is zero, apart for the pair $[k,l]=[0,0]$.

Therefore, $R[k,l] = \frac{1}{(M-k)(N-l)}\sum_{x=0}^{M-1-k}\sum_{y=0}^{N-1-l} n(x,y)n(x+k,y+l) = \sigma^2_{n(x,y)}\delta(k,l)$ where $\sigma^2_{n(x,y)}$ is the variance of noise.

Suppose now that we have L different noisy realisations of the same image $f(x,y)$ as $g_i(x,y) = f(x,y) + n_i(x,y)$, $i = 0,1,\ldots,L$. Each noise process $n_i(x,y)$ satisfies the properties (i)-(iii) given above. Moreover, $\sigma^2_{n_i(x,y)} = \sigma^2$. We form the image $\bar{g}(x,y)$ by averaging these L noisy images as follows:

$$\bar{g}(x,y) = \frac{1}{L}\sum_{i=1}^{L} g_i(x,y) = \frac{1}{L}\sum_{i=1}^{L}(f(x,y)+n_i(x,y)) = f(x,y) + \frac{1}{L}\sum_{i=1}^{L} n_i(x,y)$$

Therefore, the new image is again a noisy realisation of the original image $f(x,y)$ with noise $n(x,y) = \frac{1}{L}\sum_{i=1}^{L} n_i(x,y)$.

The mean value of the noise $n(x,y)$ is found below.

$$E\{n(x,y)\} = E\{\frac{1}{L}\sum_{i=1}^{L} n_i(x,y)\} = \frac{1}{L}\sum_{i=1}^{L} E\{n_i(x,y)\} = 0$$

The variance of the noise $n(x,y)$ is now found below.

$$\sigma^2_{n(x,y)} = E\{n^2(x,y)\} = E\left\{\left(\frac{1}{L}\sum_{i=1}^{L} n_i(x,y)\right)^2\right\} = \frac{1}{L^2}E\left\{\left(\sum_{i=1}^{L} n_i(x,y)\right)^2\right\}$$

$$= \frac{1}{L^2}E\{(\sum_{i=1}^{L} n_i^{\,2}(x,y))\} + \frac{1}{L^2}E\{(\sum_{\substack{i=1\\ i\neq j}}^{L}\sum_{j=1}^{L}(n_i(x,y)n_j(x,y))\} = \frac{1}{L^2}\sum_{i=1}^{L}E\{n_i^{\,2}(x,y)\} + \frac{1}{L^2}\sum_{\substack{i=1\\ i\neq j}}^{L}\sum_{j=1}^{L}E\{n_i(x,y)n_j(x,y)\}$$

$$= \frac{1}{L^2}\sum_{i=1}^{L}\sigma^2 + 0 = \frac{1}{L}\sigma^2$$

Therefore, we have shown that image averaging produces an image $\bar{g}(x,y)$, corrupted by noise with variance less than the variance of the noise of the original noisy images. Note that if $L \to \infty$ we have $\sigma^2_{n(x,y)} \to 0$, the resulting noise is negligible.

Recommended Questions

1. What is the importance of image enhancement in image processing? Explain in brief any two point processing techniques implemented in image processing.
2. Explain histogram equalization technique.
3. What is histogram matching? Explain the development and implementation of the method.
4. Highlight the importance of histograms in image processing and develop a procedure to perform histogram equalization.
5. Explain the following image enhancement techniques, highlighting their area of application.

i) Intensity level slicing
ii) Power – law transformation

6. Explain the following image enhancement techniques, highlighting their area of application.

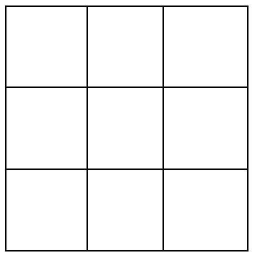
i) Bit – plane slicing.
ii) AND and OR operation

UNIT - 6

patial domain: Enhancement in the case of a single image

Spatial masks

Many image enhancement techniques are based on spatial operations performed on local neighbourhoods of input pixels. The image is usually convolved with a finite impulse response filter called spatial mask. The use of spatial masks on a digital image is called spatial filtering. Suppose that we have an image $f(x,y)$ of size N^2 and we define a neighbourhood around each pixel. For example let this neighbourhood to be a rectangular window of size 3×3



If we replace each pixel by a weighted average of its neighbourhood pixels then the response of the linear mask for the pixel z_5 is $\sum_{i=1}^{9} w_i z_i$. We may repeat the same process for the whole image.

4.2 Lowpass and highpass spatial filtering

A 3×3 spatial mask operating on an image can produce (a) a smoothed version of the image (which contains the low frequencies) or (b) it can enhance the edges and suppress essentially the constant background information. The behaviour is basically dictated by the signs of the elements of the mask.

Let us suppose that the mask has the following form

To be able to estimate the effects of the above mask with relation to the sign of the coefficients a,b,c,d,e,f,g,h, we will consider the equivalent one dimensional mask

Let us suppose that the above mask is applied to a signal $x(n)$. The output of this operation will be a signal $y(n)$ as $y(n) = dx(n-1) + x(n) + ex(n+1) \Rightarrow Y(z) = dz^{-1}X(z) + X(z) + ezX(z) \Rightarrow$

$$Y(z) = (dz^{-1} + 1 + ez)X(z) \Rightarrow \frac{Y(z)}{X(z)} = H(z) = dz^{-1} + 1 + ez \ .$$

This is the transfer function of a system that produces the above input-output relationship. In the frequency domain we have $H(e^{j\omega}) = d\exp(-j\omega) + 1 + e\exp(j\omega)$.

The values of this transfer function at frequencies $\omega = 0$ and $\omega = \pi$ are:

$$H(e^{j\omega})\Big|_{\omega=0} = d + 1 + e$$

$$H(e^{j\omega})\Big|_{\omega=\pi} = -d + 1 - e$$

If a lowpass filtering (smoothing) effect is required then the following condition must hold

$$H(e^{j\omega})\Big|_{\omega=0} \geq H(e^{j\omega})\Big|_{\omega=\pi} = d + e \geq 0$$

If a highpass filtering effect is required then

$$H(e^{j\omega})\Big|_{\omega=0} \leq H(e^{j\omega})\Big|_{\omega=\pi} = d + e \leq 0$$

The most popular masks for lowpass filtering are masks with all their coefficients positive and for highpass filtering, masks where the central pixel is positive and the surrounding pixels are negative or the other way round.

Popular techniques for lowpass spatial filtering

Uniform filtering

The most popular masks for lowpass filtering are masks with all their coefficients positive and equal to each other as for example the mask shown below. Moreover, they sum up to 1 in order to maintain the mean of the image.

$$\frac{1}{9}\times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Gaussian filtering

The two dimensional Gaussian mask has values that attempts to approximate the continuous function

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

In theory, the Gaussian distribution is non-zero everywhere, which would require an infinitely large convolution kernel, but in practice it is effectively zero more than about three standard deviations from the mean, and so we can truncate the kernel at this point. The following shows a suitable integer-valued convolution kernel that approximates a Gaussian with a σ of 1.0.

$\frac{1}{273}\times$

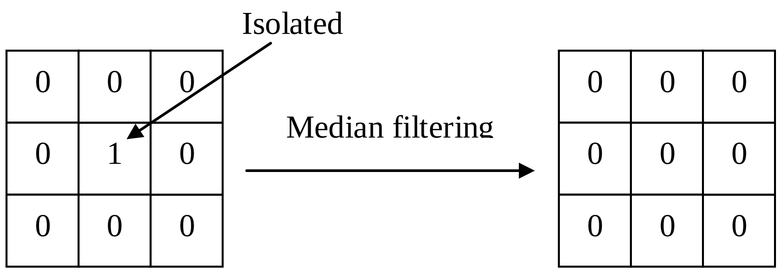
1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

Median filtering

The median m of a set of values is the value that possesses the property that half the values in the set are less than m and half are greater than m. Median filtering is the operation that replaces each pixel by the median of the grey level in the neighbourhood of that pixel.

Median filters are non linear filters because for two sequences $x(n)$ and $y(n)$ $\text{median}\{x(n)+y(n)\} \neq \text{median}\{x(n)\} + \text{median}\{y(n)\}$

Median filters are useful for removing isolated lines or points (pixels) while preserving spatial resolutions. They perform very well on images containing binary (**salt and pepper**) noise but perform poorly when the noise is Gaussian. Their performance is also poor when the number of noise pixels in the window is greater than or half the number of pixels in the window (**why?**)



Directional smoothing

To protect the edges from blurring while smoothing, a directional averaging filter can be useful. Spatial averages $g(x,y:\theta)$ are calculated in several selected directions (for example could be horizontal, vertical, main diagonals)

$$g(x,y:\theta)=\frac{1}{N_\theta}\sum_{(k,l)}\sum_{\in W_\theta} f(x-k,y-l)$$

and a direction θ^* is found such that $\left|f(x,y)-g(x,y:\theta^*)\right|$ is minimum. (Note that W_θ is the neighbourhood along the direction θ and N_θ is the number of pixels within this neighbourhood). Then by replacing $g(x,y:\theta)$ with $g(x,y:\theta^*)$ we get the desired result.

High Boost Filtering

A high pass filtered image may be computed as the difference between the original image and a lowpass filtered version of that image as follows:

(Highpass part of image) = (Original) - (Lowpass part of image)

Multiplying the original image by an amplification factor denoted by A, yields the so called high boost filter:

(Highboost image) $=(A)$ (Original)-(Lowpass) $=(A-1)$ (Original)+(Original)-(Lowpass) $=(A-1)$ (Original) + (Highpass)

The general process of subtracting a blurred image from an original as given in the first line is called ***unsharp masking***. A possible mask that implements the above procedure could be the one illustrated below.

$$\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & A & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} \;+\frac{1}{9}\times\; \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & -1 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

$$\frac{1}{9}\times\begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 9A-1 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

The high-boost filtered image looks more like the original with a degree of edge enhancement, depending on the value of A.

<u>Popular techniques for highpass spatial filtering. Edge detection using derivative filters</u>

About two dimensional high pass spatial filters

An edge is the boundary between two regions with relatively distinct grey level properties. The idea underlying most edge detection techniques is the computation of a local derivative operator. The magnitude of the first derivative calculated within a neighbourhood around the pixel of interest, can be used to detect the presence of an edge in an image.

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.

$$\nabla \underline{f}(x,y)=\begin{bmatrix}\dfrac{\partial f(x,y)}{\partial x}\\[2ex]\dfrac{\partial f(x,y)}{\partial y}\end{bmatrix}$$

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

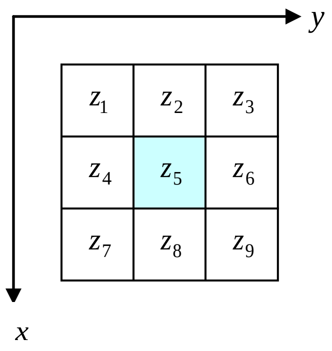
$$\nabla f(x,y)=\mathrm{mag}(\nabla \underline{f}(x,y))=\left[\left(\frac{\partial f(x.y)}{\partial x}\right)^2+\left(\frac{\partial f(x,y)}{\partial y}\right)^2\right]^{1/2}$$

Common practice is to approximate the gradient with absolute values which is simpler to implement as follows.

$$\nabla f(x,y)\cong\left|\frac{\partial f(x,y)}{\partial x}\right|+\left|\frac{\partial f(x,y)}{\partial y}\right|$$

(1) Consider a pixel of interest $f(x, y) = z_5$ and a rectangular neighbourhood of size

$3 \times 3 = 9$ pixels (including the pixel of interest) as shown below.



Roberts operator

Equation (1) can be approximated at point z_5 in a number of ways. The simplest is to use the difference $(z_5 - z_8)$ in the x direction and $(z_5 - z_6)$ in the y direction. This approximation is known as the **Roberts** operator, and is expressed mathematically as follows.

$$\nabla f \cong |z_5 - z_8| + |z_5 - z_6|$$

(2) Another approach for approximating (1) is to use cross differences

$$\nabla f \cong |z_5 - z_9| + |z_6 - z_8|$$

(3) Equations (2), (3) can be implemented by using the following masks. The original image is convolved with both masks separately and the absolute values of the two outputs of the convolutions are added.

1	0
-1	0

1	-1
0	0

Roberts operator

1	0
0	-1

0	1
-1	0

Roberts operator

Prewitt operator

Another approximation of equation (1) but using now a 3×3 mask is the following.

$$\nabla f \cong \left|(z_7+z_8+z_9)-(z_1+z_2+z_3)\right|+\left|(z_3+z_6+z_9)-(z_1+z_4+z_7)\right|$$
(4)

This approximation is known as the **Prewitt** operator. Equation (4) can be implemented by using the following masks. Again, the original image is convolved with both masks separately and the absolute values of the two outputs of the convolutions are added.

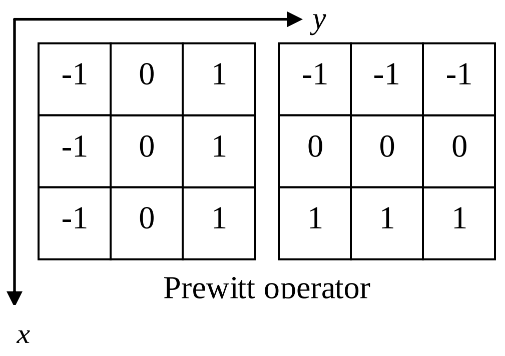
Sobel operator.

Definition and comparison with the Prewitt operator

The most popular approximation of equation (1) but using a 3×3 mask is the following.

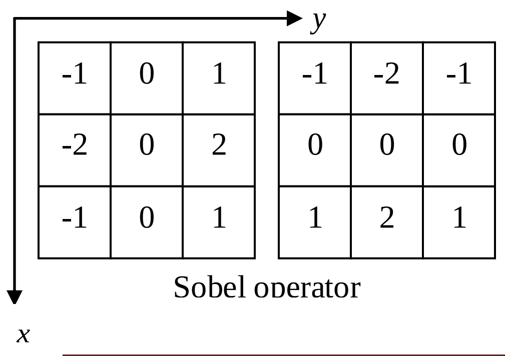
$$\nabla f \cong \left|(z_7+2z_8+z_9)-(z_1+2z_2+z_3)\right|+\left|(z_3+2z_6+z_9)-(z_1+2z_4+z_7)\right|$$
(5)

This approximation is known as the **Sobel** operator.



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

Prewitt operator



-1	0	1		-1	-2	-1
-2	0	2		0	0	0
-1	0	1		1	2	1

Sobel operator

If we consider the left mask of the Sobel operator, this causes differentiation along the y

direction. A question that arises is the following: What is the effect caused by the same

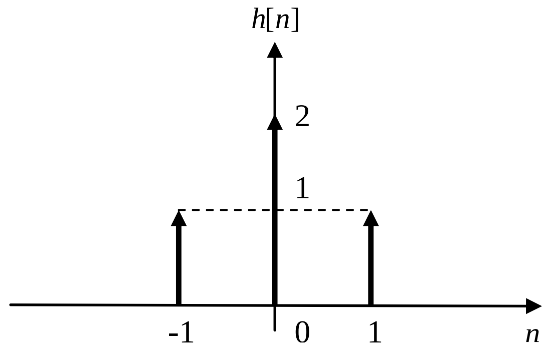
mask along the x direction?

If we isolate the following part of the mask

1
2
1

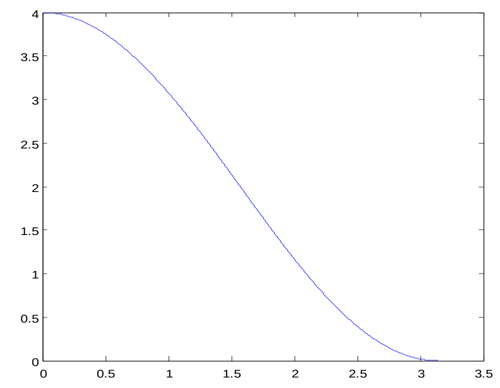
and treat it as a one dimensional mask, we are interested in finding the effects of that mask. We will therefore, treat this mask as a one dimensional impulse response $h[n]$ of the form

or
$$h[n] = \begin{cases} 1 & n = -1 \\ 2 & n = 0 \\ 1 & n = 1 \\ 0 & \text{otherwise} \end{cases}$$



The above response applied to a signal $x[n]$ yields a signal $y[n] = x[n-1] + 2x[n] + x[n+1]$ or in z-transform domain $Y(z) = (z^{-1} + 2 + z)X(z) \Rightarrow Y(j\omega) = 2(\cos\omega + 1)X(j\omega)$. Therefore,

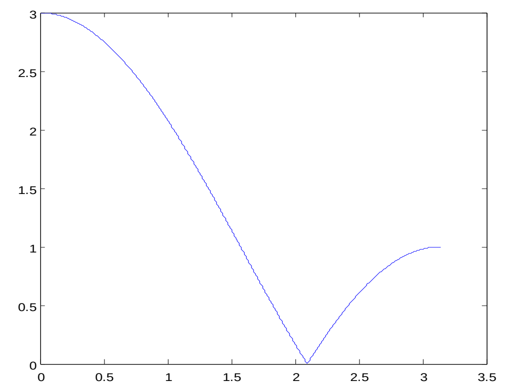
$h[n]$ is the impulse response of a system with transfer function $H(j\omega) = 2(\cos\omega + 1) = |H(j\omega)|$ shown in the figure below for $[0, \pi]$. This is a lowpass filter type of response. Therefore, we can claim that the Sobel operator has a differentiation effect along one of the two directions and a smoothing effect along the other direction.



The same analysis for the Prewitt operator would give

$$Y(z) = (z^{-1} + 1 + z)X(z)$$
$$\Rightarrow Y(j\omega) = (2\cos\omega + 1)X(j\omega) \Rightarrow |H(j\omega)| = |2\cos\omega + 1|$$

shown in the figure below for $[0, \pi]$. This response looks "strange" since it decreases up to the point $|2\cos\omega + 1| = 0 \Rightarrow \cos\omega = -0.5$ and then starts increasing.



Based on the above analysis it is stated in the literature that the Sobel operator have the advantage of providing both a differencing a smoothing effect while Prewitt does not. However, if you implement both operators you cannot see any visual difference.

Laplacian operator

The **Laplacian** of a 2-D function $f(x,y)$ is a second order derivative defined as

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

In practice it can be also implemented using a 3x3 mask as follows (**why?**)

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8)$$

The main disadvantage of the Laplacian operator is that it produces double edges (**why?**).

<u>Recommended Questions</u>

1. Explain the smoothing of images in frequency domain using:

i) Ideal low pass filter

ii) Butterworth lowpass filter

2. With a block diagram and equations, explain the homomorphic filtering. How dynamic range compression and contrast enhancement is simultaneously achieved?
3. Discuss homomorphic filtering.
4. Explain sharpening filters in the frequency domain
5. Explain the basic concept of spatial filtering in image enhancement and hence explain the importance of smoothing filters and median filters.

Unit-7

Preliminaries

What is image restoration?

Image Restoration refers to a class of methods that aim to remove or reduce the degradations that have occurred while the digital image was being obtained. All natural images when displayed have gone through some sort of degradation:

- during display mode
- during acquisition mode, or
- during processing mode

The degradations may be due to

- sensor noise
- blur due to camera misfocus
- relative object-camera motion
- random atmospheric turbulence
- others

In most of the existing image restoration methods we assume that the degradation process can be described using a mathematical model.

How well can we do?

Depends on how much we know about

- the original image
- the degradations

(how accurate our models are)

Image restoration and image enhancement differences

- Image restoration differs from image enhancement in that the latter is concerned more with accentuation or extraction of image features rather than restoration of degradations.

- Image restoration problems can be quantified precisely, whereas enhancement criteria are difficult to represent mathematically.

Image observation models

Typical parts of an imaging system: image formation system, a detector and a recorder. A general model for such a system could be:

$$y(i,j) = r\left[w(i,j)\right] + n(i,j)$$

$$w(i,j) = H\left[f(i,j)\right] = \iint_{-\infty}^{\infty} h(i,j,i',j') f(i',j') di'dj'$$

$$n(i,j) = g\left(r\{w(i,j)\}\right) n_1(i,j) + n_2(i,j)$$

Where $y(i,j)$ is the degraded image, $f(i,j)$ is the original image and $h(i,j,i',j')$ is an operator that represents the degradation process, for example a blurring process. Functions $g(\cdot)$ and $r(\cdot)$ are generally nonlinear, and represent the characteristics of detector/recording mechanisms. $n(i,j)$ is the additive noise, which has an image-dependent random component $g\left(r\{H[f(i,j)]\}\right) n_1(i,j)$ and an image-independent random component $n_2(i,j)$.

Detector and recorder models

The response of image detectors and recorders in general is nonlinear. An example is the response of image scanners

$$r(i,j) = \alpha w(i,j)^{\beta}$$

where α and β are device-dependent constants and $w(i,j)$ is the input blurred image.
For photofilms

$$r(i,j) = \gamma \log_{10} w(i,j) - r_0$$

[

where γ is called the gamma of the film, $w(i,j)$ is the incident light intensity and $r(i,j)$ is called the optical density. A film is called positive if it has negative γ.

Noise models

The general noise model

$$n(i,j) = g\{r[w(i,j)]\} n_1(i,j) + n_2(i,j)$$

is applicable in many situations. Example, in photoelectronic systems we may have
$g(x) = \sqrt{x}$. Therefore,

$$n(i,j) = \sqrt{\alpha w(i,j)^{\beta}}\, n_1(i,j) + n_2(i,j)$$

where n_1 and n_2 are zero-mean, mutually independent, Gaussian white noise fields. The term $n_2(i,j)$ may be referred as thermal noise. In the case of films there is no thermal noise and the noise model is

$$n(i,j) = \sqrt{\gamma \log_{10} w(i,j) - r_o}\, n_1(i,j)$$

Because of the signal-dependent term in the noise model, restoration algorithms are quite
difficult. Often $w(i,j)$ is replaced by its spatial average, μ_w, giving

$$n(i,j) = g\left[r\left[\mu_w\right]\right] n_1(i,j) + n_2(i,j)$$

which makes $n(i,j)$ a Gaussian white noise random field. A lineal observation model for
photoelectronic devices is $y(i,j) = w(i,j) + \sqrt{\mu_w}\, n_1(i,j) + n_2(i,j)$

For photographic films with $\gamma = -1$

$$y(i,j) = -\log_{10} w(i,j) - r_0 + a n_1(x,y)$$

where r_0, a are constants and r_0 can be ignored.

The light intensity associated with the observed optical density $y(i,j)$ is

$$I(i,j) = 10^{-y(i,j)} = w(i,j)10^{-an_1(i,j)} = w(i,j)n(i,j)$$

where $n(i,j) \hat{=} 10^{-an_1(i,j)}$ now appears as multiplicative noise having a log-normal distribution.

Keep in mind that we are just referring to the most popular image observation models. In the literature you can find a quite large number of different image observation models. Image restoration algorithms are based on the above image formation models.

A general model of a simplified digital image degradation process

A simplified version for the image restoration process model is

$$y(i,j) = H[f(i,j)] + n(i,j)$$

Where $y(i,j)$ the degraded image

$f(i,j)$ the original image

H an operator that represents the degradation process

$n(i,j)$ the external noise which is assumed to be image-independent

Possible classification of restoration methods

Restoration methods could be classified as follows:

- **deterministic:** we work with sample by sample processing of the observed (degraded) image
- **stochastic :** we work with the statistics of the images involved in the process
- **non-blind :** the degradation process H is known
- **blind : t**he degradation process H is unknown
- **semi-blind :** the degradation process H could be considered partly known

From the viewpoint of implementation:

- **direct**
- **iterative**
- **recursive**

<u>**Linear position invariant degradation models**</u>

Definition

We again consider the general degradation model

$$y(i,j) = H[f(i,j)] + n(i,j)$$

If we ignore the presence of the external noise $n(i,j)$ we get

$$y(i,j) = H[f(i,j)]$$

H is ***linear*** if

$$H[k_1 f_1(i,j) + k_2 f_2(i,j)] = k_1 H[f_1(i,j)] + k_2 H[f_2(i,j)]$$

H is ***position*** (or ***space***) ***invariant*** if

$$H[f(i-a, j-b)] = y(i-a, j-b)$$

From now on we will deal with linear, space invariant type of degradations.

In a real life problem many types of degradations can be approximated by linear, position invariant processes.

Advantage: Extensive tools of linear system theory become available.

Disadvantage: In some real life problems ***nonlinear*** and ***space variant*** models would be more appropriate for the description of the degradation phenomenon.

Typical linear position invariant degradation models

- **Motion blur**. It occurs when there is relative motion between the object and the camera during exposure.

$$h(i) = \begin{cases} \frac{1}{L}, & \text{if } -\frac{L}{2} \le i \le \frac{L}{2} \\ 0, & \text{otherwise} \end{cases}$$

- **Atmospheric turbulence.** It is due to random variations in the reflective index of the medium between the object and the imaging system and it occurs in the imaging of astronomical objects.

$$h(i, j) = K \exp\left(-\frac{i^2 + j^2}{2\sigma^2}\right)$$

- **Uniform out of focus blur**

$$h(i, j) = \begin{cases} \frac{1}{\pi R}, & \text{if } \sqrt{i^2 + j^2} \le R \\ 0, & \text{otherwise} \end{cases}$$

- **Uniform 2-D blur**

$$h(i, j) = \begin{cases} \frac{1}{(L)^2}, & \text{if } -\frac{L}{2} \le i, j \le \frac{L}{2} \\ 0, & \text{otherwise} \end{cases}$$

Some characteristic metrics for degradation models

- **Blurred Signal-to-Noise Ratio (BSNR)**: a metric that describes the degradation model.

$$\mathrm{BSNR} = 10\log_{10}\left\{\frac{\frac{1}{MN}\sum_i\sum_j \left[g(i,j)-\bar{g}(i,j)\right]^2}{\sigma_n^2}\right\}$$

$g(i,j) = y(i,j) - n(i,j)$

$\bar{g}(i,j) = E\{g(i,j)\}$

[

σ_n^2 : variance of additive noise

- **Improvement in SNR (ISNR)**: validates the performance of the image restoration algorithm.

$$\mathrm{ISNR} = 10\log_{10}\left\{\frac{\sum_i\sum_j \left[f(i,j)-y(i,j)\right]^2}{\sum_i\sum_j \left[f(i,j)-\hat{f}(i,j)\right]^2}\right\}$$

where $\hat{f}(i,j)$ is the restored image.

Both BSNR and ISNR can only be used for simulation with artificial data.

One dimensional discrete degradation model. Circular convolution

Suppose we have a one-dimensional discrete signal $f(i)$ of size A samples $f(0), f(1), \ldots, f(A-1)$, which is due to a degradation process. The degradation can be modeled by a one-dimensional discrete impulse response $h(i)$ of size B samples. If we assume that the degradation is a causal function we have the samples $h(0), h(1), \ldots, h(B-1)$ .We form the extended versions of $f(i)$ and $h(i)$, both of size $M \geq A+B-1$ and periodic with period M. These can be denoted as $f_e(i)$ and $h_e(i)$. For a time invariant degradation process we obtain the discrete convolution formulation as follows

$$y_e(i) = \sum_{m=0}^{M-1} f_e(m)h_e(i-m) + n_e(i)$$

Using matrix notation we can write the following form

$$\mathbf{y} = \mathbf{Hf} + \mathbf{n} \qquad \mathbf{f} = \begin{bmatrix} f_e(0) \\ f_e(1) \\ \vdots \\ f_e(M-1) \end{bmatrix},$$

$$\underset{(M\times M)}{\mathbf{H}} = \begin{bmatrix} h_e(0) & h_e(-1) & \dots & h_e(-M+1) \\ h_e(1) & h_e(0) & \dots & h_e(-M+2) \\ \vdots & \vdots & \ddots & \vdots \\ h_e(M-1) & h_e(M-2) & \dots & h_e(0) \end{bmatrix}$$

At the moment we decide to ignore the external noise $\mathbf{n}$. Because h is periodic with period M we have that

$$\underset{(M\times M)}{\mathbf{H}} = \begin{bmatrix} h_e(0) & h_e(M-1) & \dots & h_e(1) \\ h_e(1) & h_e(0) & \dots & h_e(2) \\ \vdots & \vdots & \ddots & \vdots \\ h_e(M-1) & h_e(M-2) & \dots & h_e(0) \end{bmatrix}$$

We define $\lambda(k)$ to be

$$\lambda(k) = h_e(0) + h_e(M-1)\exp(j\frac{2\pi}{M}k) + h_e(M-2)\exp(j\frac{2\pi}{M}2k) + \dots$$
$$+ h_e(1)\exp[j\frac{2\pi}{M}(M-1)k], \quad k = 0,1,\dots,M-1$$

Because $\exp[j\frac{2\pi}{M}(M-i)k] = \exp(-j\frac{2\pi}{M}ik)$ we have that

$$\lambda(k) = MH(k)$$

$H(k)$ is the discrete Fourier transform of $h_e(i)$.

I define $\mathbf{w}(k)$ to be

$$\mathbf{w}(k) = \begin{bmatrix} 1 \\ \exp(j\frac{2\pi}{M}k) \\ \vdots \\ \exp[j\frac{2\pi}{M}(M-1)k] \end{bmatrix}$$

It can be seen that

$$\mathbf{Hw}(k) = \lambda(k)\mathbf{w}(k)$$

This implies that $\lambda(k)$ is an eigenvalue of the matrix $\mathbf{H}$ and $\mathbf{w}(k)$ is its corresponding eigenvector.

We form a matrix $\mathbf{w}$ whose columns are the eigenvectors of the matrix $\mathbf{H}$, that is to say

$$\mathbf{W} = [\mathbf{w}(0) \quad \mathbf{w}(1) \quad \dots \quad \mathbf{w}(M-1)]$$

$$w(k,i) = \exp\left[j\frac{2\pi}{M}ki\right] \text{ and } w^{-1}(k,i) = \frac{1}{M}\exp\left[-j\frac{2\pi}{M}ki\right]$$

We can then diagonalize the matrix $\mathbf{H}$ as follows

$$\mathbf{H} = \mathbf{WDW}^{-1} \Rightarrow \mathbf{D} = \mathbf{W}^{-1}\mathbf{HW}$$

where

$$\mathbf{D} = \begin{bmatrix} \lambda(0) & & \mathbf{0} & \\ & \lambda(1) & & \\ & & \ddots & \\ \mathbf{0} & & & \lambda(M-1) \end{bmatrix}$$

Obviously $\mathbf{D}$ is a diagonal matrix and

$$D(k,k) = \lambda(k) = MH(k)$$

If we go back to the degradation model we can write

$$\mathbf{y} = \mathbf{Hf} \Rightarrow \mathbf{y} = \mathbf{WDW}^{-1}\mathbf{f} \Rightarrow \mathbf{W}^{-1}\mathbf{y} = \mathbf{DW}^{-1}\mathbf{f} \Rightarrow Y(k) = MH(k)F(k), k = 0,1,\ldots,M-1$$

$Y(k), H(k), F(k), k = 0,1,\ldots,M-1$ are the $M-$ sample discrete Fourier transforms of $y(i), h(i), f(i)$, respectively. So by choosing $\lambda(k)$ and $\mathbf{w}(k)$ as above and assuming that $h_e(i)$ is periodic, we start with a matrix problem and end up with M scalar problems.

Two dimensional discrete degradation model. Circular convolution

Suppose we have a two-dimensional discrete signal $f(i,j)$ of size $A \times B$ samples which is due to a degradation process. The degradation can now be modeled by a two dimensional discrete impulse response $h(i,j)$ of size $C \times D$ samples. We form the extended versions of $f(i,j)$ and $h(i,j)$, both of size $M \times N$, where $M \geq A + C - 1$ and $N \geq B + D - 1$, and periodic with period $M \times N$. These can be denoted as $f_e(i,j)$ and $h_e(i,j)$. For a space invariant degradation process we obtain

$$y_e(i,j) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f_e(m,n)h_e(i-m, j-n) + n_e(i,j)$$

Using matrix notation we can write the following form

$$\mathbf{y} = \mathbf{Hf} + \mathbf{n}$$

where $\mathbf{f}$ and $\mathbf{y}$ are $MN-$ dimensional column vectors that represent the lexicographic

ordering of images $f_e(i,j)$ and $h_e(i,j)$ respectively.

$$\mathbf{H} = \begin{bmatrix} \mathbf{H_0} & \mathbf{H_{M-1}} & \cdots & \mathbf{H_1} \\ \mathbf{H_1} & \mathbf{H_0} & \cdots & \mathbf{H_2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{H_{M-1}} & \mathbf{H_{M-2}} & \cdots & \mathbf{H_0} \end{bmatrix}$$

$$\mathbf{H}_j = \begin{bmatrix} h_e(j,0) & h_e(j,N-1) & \dots & h_e(j,1) \\ h_e(j,1) & h_e(j,0) & \dots & h_e(j,2) \\ \vdots & \vdots & \ddots & \vdots \\ h_e(j,N-1) & h_e(j,N-2) & \dots & h_e(j,0) \end{bmatrix}$$

The analysis of the diagonalisation of $\mathbf{H}$ is a straightforward extension of the one-

dimensional case. In that case we end up with the following set of $M \times N$ scalar problems.

$$Y(u,v) = MNH(u,v)F(u,v)(+N(u,v))$$

$$u = 0,1,\dots,M-1, v = 0,1,\dots,N-1$$

In the general case we may have two functions $f(i), A \le i \le B$ and $h(i), C \le i \le D$, where A, C can be also negative (in that case the functions are **non-causal**). For the periodic convolution we have to extend the functions from both sides knowing that the convolution is $g(i) = h(i) * f(i), A + C \le i \le B + D$.

Direct deterministic approaches to restoration

Inverse filtering

The objective is to minimize

$$J(\mathbf{f}) = \|\mathbf{n}(\mathbf{f})\|^2 = \|\mathbf{y} - \mathbf{Hf}\|^2$$

We set the first derivative of the cost function equal to zero

$$\frac{\partial J(\mathbf{f})}{\partial \mathbf{f}} = 0 \Rightarrow -2\mathbf{H^T}(\mathbf{y} - \mathbf{Hf}) = \mathbf{0}$$

$$\mathbf{f} = (\mathbf{H^T H})^{\mathbf{-1}} \mathbf{H^T y}$$

If $M = N$ and $\mathbf{H}^{-1}$ exists then

$$\mathbf{f} = \mathbf{H^{-1} y}$$

According to the previous analysis if $\mathbf{H}$ (and therefore $\mathbf{H^{-1}}$) is block circulant the above

[
problem can be solved as a set of $M \times N$ scalar problems as follows

$$F(u,v) = \frac{H^*(u,v)Y(u,v)}{|H(u,v)|^2} \Rightarrow f(i,j) = \Im^{-1}\left[\frac{H^*(u,v)Y(u,v)}{|H(u,v)|^2}\right]$$

Computational issues concerning inverse filtering

(I) Suppose first that the additive noise $n(i,j)$ is negligible. A problem arises if $H(u,v)$ becomes very small or zero for some point (u,v) or for a whole region in the (u,v) plane. In that region inverse filtering cannot be applied. Note that in most real applications $H(u,v)$ drops off rapidly as a function of distance from the origin. The solution is that if these points are known they can be neglected in the computation of $F(u,v)$.

(II) In the presence of external noise we have that

$$\hat{F}(u,v) = \frac{H^*(u,v)\left(Y(u,v) + N(u,v)\right)}{|H(u,v)|^2} = \frac{H^*(u,v)Y(u,v)}{|H(u,v)|^2} + \frac{H^*(u,v)N(u,v)}{|H(u,v)|^2} \Rightarrow$$

$$\hat{F}(u,v) = F(u,v) + \frac{N(u,v)}{H(u,v)}$$

If $H(u,v)$ becomes very small, the term $N(u,v)$ dominates the result. The solution is again to carry out the restoration process in a limited neighborhood about the origin where $H(u,v)$ is not very small. This procedure is called **pseudoinverse filtering**. In that case we set

$$\hat{F}(u,v) = \begin{cases} \dfrac{H^*(u,v)Y(u,v)}{|H(u,v)|^2} & H(u,v) \geq T \\ 0 & H(u,v) < T \end{cases}$$

The threshold T is defined by the user. In general, the noise may very well possess large components at high frequencies (u,v), while $H(u,v)$ and $Y(u,v)$ normally will be dominated by low frequency components.

Constrained least squares (CLS) restoration

It refers to a very large number of restoration algorithms.The problem can be formulated as follows.

minimize

$$J(\mathbf{f}) = \|\mathbf{n}(\mathbf{f})\|^2 = \|\mathbf{y} - \mathbf{Hf}\|^2$$

subject to

$$\|\mathbf{Cf}\|^2 < \varepsilon$$

where $\mathbf{Cf}$ is a high pass filtered version of the image. **The idea behind the above constraint is that the highpass version of the image contains a considerably large amount of noise!** Algorithms of the above type can be handled using optimization techniques. Constrained least squares (CLS) restoration can be formulated by choosing an $\mathbf{f}$ to minimize the Lagrangian

$$\min\left(\|\mathbf{y} - \mathbf{Hf}\|^2 + \alpha\|\mathbf{Cf}\|^2\right)$$

Typical choice for $\mathbf{C}$ is the 2-D Laplacian operator given by

$$\mathbf{C} = \begin{bmatrix} 0.00 & -0.25 & 0.00 \\ -0.25 & 1.00 & -0.25 \\ 0.00 & -0.25 & 0.00 \end{bmatrix}$$

α represents either a Lagrange multiplier or a fixed parameter known as **regularisation parameter** and it controls the relative contribution between the term $\|\mathbf{y} - \mathbf{Hf}\|^2$ and the term $\|\mathbf{Cf}\|^2$. The minimization of the above leads to the following estimate for the original image

$$\mathbf{f} = \left(\mathbf{H}^{\mathbf{T}}\mathbf{H} + \alpha\mathbf{C}^{\mathbf{T}}\mathbf{C}\right)^{-1}\mathbf{H}^{\mathbf{T}}\mathbf{y}$$

Computational issues concerning the CLS method

Choice of α

The problem of the choice of α has been attempted in a large number of studies and different techniques have been proposed. One possible choice is based on a **set theoretic approach**: a restored image is approximated by an image which lies in the intersection of the two ellipsoids defined by

$$Q_{\mathbf{f}|\mathbf{y}} = \{\mathbf{f} \mid \|\mathbf{y} - \mathbf{Hf}\|^2 \le E^2\} \text{ and}$$

$$Q_{\mathbf{f}} = \{\mathbf{f} \mid \|\mathbf{Cf}\|^2 \le \varepsilon^2\}$$

The center of one of the ellipsoids which bounds the intersection of $Q_{\mathbf{f}|\mathbf{y}}$ and $Q_{\mathbf{f}}$, is given by the equation

$$\mathbf{f} = \left(\mathbf{H}^T\mathbf{H} + \alpha\mathbf{C}^T\mathbf{C}\right)^{-1}\mathbf{H}^T\mathbf{y}$$

with $\alpha = (E/\varepsilon)^2$. Another problem is then the choice of E^2 and ε^2. One choice

[

could be

$$\alpha = \frac{1}{\text{BSNR}}$$

Comments

With larger values of α, and thus more regularisation, the restored image tends to have more ***ringing***. With smaller values of α, the restored image tends to have more ***amplified noise effects***. The variance and bias of the error image in frequency domain are

$$Var(\alpha) = \sigma_n^2 \sum_{u=0}^{M}\sum_{v=0}^{N} \frac{|H(u,v)|^2}{\left(|H(u,v)|^2 + \alpha|C(u,v)|^2\right)^2}$$

$$Bias(\alpha) = \sigma_n^2 \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} \frac{|F(u,v)|^2\alpha^2|C(u,v)|^4}{\left(|H(u,v)|^2 + \alpha|C(u,v)|^2\right)^2}$$

The minimum MSE is encountered close to the intersection of the above functions. A good choice of α is one that gives the best compromise between the variance and bias of the error image.

Iterative deterministic approaches to restoration

They refer to a large class of methods that have been investigated extensively over the last decades. They possess the following advantages.

- There is no need to explicitly implement the inverse of an operator. The restoration process is monitored as it progresses. Termination of the algorithm may take place before convergence.
- The effects of noise can be controlled in each iteration.
- The algorithms used can be spatially adaptive.
- The problem specifications are very flexible with respect to the type of degradation. Iterative techniques can be applied in cases of spatially varying or nonlinear degradations or in cases where the type of degradation is completely unknown (blind restoration).

In general, iterative restoration refers to any technique that attempts to minimize a function of the form $M(\mathbf{f})$ using an updating rule for the partially restored image.

Least squares iteration

In that case we seek for a solution that minimizes the function

$$M(\mathbf{f}) = \|\mathbf{y} - \mathbf{Hf}\|^{\mathbf{2}}$$

A necessary condition for $M(\mathbf{f})$ to have a minimum is that its gradient with respect to $\mathbf{f}$ is equal to zero. This gradient is given below

$$\frac{\partial M(\mathbf{f})}{\partial \mathbf{f}} = \nabla_{\mathbf{f}} M(\mathbf{f}) = 2(-\mathbf{H}^{\mathbf{T}}\mathbf{y} + \mathbf{H}^{\mathbf{T}}\mathbf{Hf})$$

and by using the steepest descent type of optimization we can formulate an iterative rule as follows:

$$\mathbf{f_0} = \beta \mathbf{H}^{\mathbf{T}}\mathbf{y}$$

$$\mathbf{f_{k+1}} = \mathbf{f_k} - \mu \frac{\partial M(\mathbf{f_k})}{\partial \mathbf{f_k}} = \mathbf{f_k} + \beta \mathbf{H}^{\mathbf{T}}(\mathbf{y} - \mathbf{Hf_k}) = \beta \mathbf{H}^{\mathbf{T}}\mathbf{y} + (\mathbf{I} - \beta \mathbf{H}^{\mathbf{T}}\mathbf{H})\mathbf{f_k}$$

Constrained least squares iteration

In this method we attempt to solve the problem of constrained restoration iteratively. As already mentioned the following functional is minimized

$$M(\mathbf{f}, \alpha) = \|\mathbf{y} - \mathbf{Hf}\|^2 + \alpha \|\mathbf{Cf}\|^2$$

The necessary condition for a minimum is that the gradient of $M(\mathbf{f}, \alpha)$ is equal to zero. That gradient is

$$\Phi(\mathbf{f}) = \nabla_{\mathbf{f}} M(\mathbf{f}, \alpha) = 2[(\mathbf{H}^{\mathbf{T}}\mathbf{H} + \alpha \mathbf{C}^{\mathbf{T}}\mathbf{C})\mathbf{f} - \mathbf{H}^{\mathbf{T}}\mathbf{y}]$$

The initial estimate and the updating rule for obtaining the restored image are now given by

$$\mathbf{f_0} = \beta \mathbf{H}^{\mathbf{T}}\mathbf{y}$$

$$\mathbf{f_{k+1}} = \mathbf{f_k} + \beta[\mathbf{H}^{\mathbf{T}}\mathbf{y} - (\mathbf{H}^{\mathbf{T}}\mathbf{H} + \alpha \mathbf{C}^{\mathbf{T}}\mathbf{C})\mathbf{f_k}]$$

It can be proved that the above iteration (known as **Iterative CLS** or **Tikhonov-Miller Method**) converges if

$$0 < \beta < \frac{2}{|\lambda_{\max}|}$$

where $\lambda_{\max}$ is the maximum eigenvalue of the matrix

$$(\mathbf{H}^{\mathbf{T}}\mathbf{H} + \alpha \mathbf{C}^{\mathbf{T}}\mathbf{C})$$

If the matrices $\mathbf{H}$ and $\mathbf{C}$ are block-circulant the iteration can be implemented in the frequency domain.

Projection onto convex sets (POCS)

The set-based approach described previously can be generalized so that any number of prior constraints can be imposed as long as the constraint sets are closed convex. If the constraint sets have a non-empty intersection, then a solution that belongs to the intersection set can be found by the method of POCS. Any solution in the intersection set is consistent with the a priori constraints and therefore it is a feasible solution.

Let $Q_1, Q_2, \ldots, Q_m$ be closed convex sets in a finite dimensional vector space, with $P_1, P_2, \ldots, P_m$ their respective projectors. The iterative procedure

$$\mathbf{f_{k+1}} = P_1 P_2, \ldots P_m \mathbf{f_k}$$

converges to a vector that belongs to the intersection of the sets $Q_i, i = 1,2,\ldots,m$, for any starting vector $\mathbf{f_0}$. An iteration of the form $\mathbf{f_{k+1}} = P_1 P_2 \mathbf{f_k}$ can be applied in the problem described previously, where we seek for an image which lies in the intersection of the two ellipsoids defined by

$$Q_{\mathbf{f|y}} = \{\mathbf{f} \mid \|\mathbf{y} - \mathbf{Hf}\|^2 \le E^2\} \text{ and } Q_{\mathbf{f}} = \{\mathbf{f} \mid \|\mathbf{Cf}\|^2 \le \varepsilon^2\}$$

The respective projections $P_1\mathbf{f}$ and $P_2\mathbf{f}$ are defined by

$$P_1\mathbf{f} = \mathbf{f} + \lambda_1 (\mathbf{I} + \lambda_1 \mathbf{H^T H})^{-1} \mathbf{H^T}(\mathbf{y} - \mathbf{Hf})$$

$$P_2\mathbf{f} = [\mathbf{I} - \lambda_2 (\mathbf{I} + \lambda_2 \mathbf{C^T C})^{-1} \mathbf{C^T C}]\mathbf{f}$$

[

Spatially adaptive iteration

The functional to be minimized takes the form

$$M(\mathbf{f}, \alpha) = \|\mathbf{y} - \mathbf{Hf}\|^2_{\mathbf{W_1}} + \alpha \|\mathbf{Cf}\|^2_{\mathbf{W_2}}$$

where

$$\|\mathbf{y} - \mathbf{Hf}\|^2_{\mathbf{W_1}} = (\mathbf{y} - \mathbf{Hf})^{\mathbf{T}} \mathbf{W_1} (\mathbf{y} - \mathbf{Hf})$$

$$\|\mathbf{Cf}\|^2_{\mathbf{W_2}} = (\mathbf{Cf})^{\mathbf{T}} \mathbf{W_2} (\mathbf{Cf})$$

$\mathbf{W_1}, \mathbf{W_2}$ are diagonal matrices, the choice of which can be justified in various ways. The entries in both matrices are non-negative values and less than or equal to unity. In that case

$$\Phi(\mathbf{f}) = \nabla_{\mathbf{f}} M(\mathbf{f}, \alpha) = (\mathbf{H^T W_1 H} + \alpha \mathbf{C^T W_2 C})\mathbf{f} - \mathbf{H^T W_1 y}$$

A more specific case is

$$M(\mathbf{f}, \alpha) = \|\mathbf{y} - \mathbf{Hf}\|^2 + \alpha \|\mathbf{Cf}\|^2_{\mathbf{W}}$$

where the weighting matrix is incorporated only in the regularization term. This method is known as **weighted regularised image restoration**. The entries in matrix $\mathbf{W}$ will be chosen so that the high-pass filter is only effective in the areas of low activity and a very little smoothing takes place in the edge areas.

Robust functionals

Robust functionals allow for the efficient supression of a wide variety of noise processes and permit the reconstruction of sharper edges than their quadratic counterparts. We are seeking to minimize

$$M(\mathbf{f},\alpha) = R_n(\mathbf{y} - \mathbf{Hf}) + \alpha R_x \mathbf{Cf}$$

$R_n(), R_x()$ are referred to as **residual** and **stabilizing** functionals respectively.

Computational issues concerning iterative techniques

(I) Convergence

The **contraction mapping theorem** usually serves as a basis for establishing convergence of iterative algorithms. According to it iteration

$$\mathbf{f_0} = \mathbf{0}$$
$$\mathbf{f_{k+1}} = \mathbf{f_k} + \beta\Phi(\mathbf{f_k}) = \Psi(\mathbf{f_k})$$

converges to a unique fixed point $\mathbf{f}^*$, that is, a point such that $\Psi(\mathbf{f}^*) = \mathbf{f}^*$, for any

initial vector, if the operator or transformation $\Psi(\mathbf{f})$ is a **contraction**. This means

that for any two vectors $\mathbf{f_1}$ and $\mathbf{f_2}$ in the domain of $\Psi(\mathbf{f})$ the following relation

holds

$$\|\Psi(\mathbf{f_1}) - \Psi(\mathbf{f_2})\| \le \eta \|\mathbf{f_1} - \mathbf{f_2}\|$$

with $\eta \le 1$ and $\|\cdot\|$ any norm. The above condition is **norm dependent.**

(II) Rate of convergence

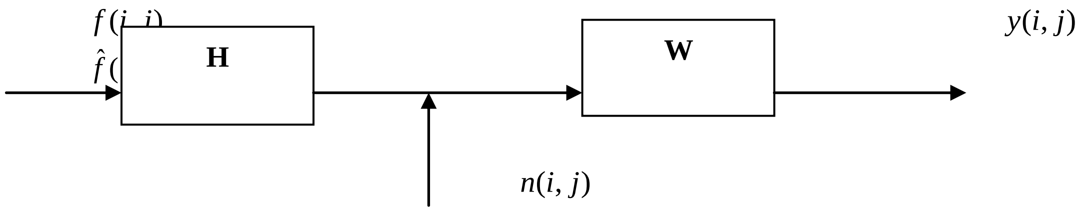
The termination criterion most frequently used compares the normalized change in energy at each iteration to a threshold such as

$$\frac{\|\mathbf{f_{k+1}}-\mathbf{f_k}\|^2}{\|\mathbf{f_k}\|^2}\le 10^{-6}$$

Stochastic approaches to restoration

Wiener estimator (stochastic regularisation)

The image restoration problem can be viewed as a system identification problem as follows:



The objective is to minimize the following function

$$E\{(\mathbf{f}-\hat{\mathbf{f}})^{\mathbf{T}}(\mathbf{f}-\hat{\mathbf{f}})\}$$

To do so the following conditions should hold:

(i) $E\{\hat{\mathbf{f}}\}=E\{\mathbf{f}\}\Rightarrow E\{\mathbf{f}\}=\mathbf{W}E\{\mathbf{y}\}$

(ii) The error must be orthogonal to the observation about the mean

$E\{(\hat{\mathbf{f}}-\mathbf{f})(\mathbf{y}-E\{\mathbf{y}\})^{\mathbf{T}}\}=0$

From (i) and (ii) we have that

$$E\{(\mathbf{Wy}-\mathbf{f})(\mathbf{y}-E\{\mathbf{y}\})^{\mathbf{T}}\}=0\Rightarrow E\{(\mathbf{Wy}+E\{\mathbf{f}\}-\mathbf{W}E\{\mathbf{y}\}-\mathbf{f})(\mathbf{y}-E\{\mathbf{y}\})^{\mathbf{T}}\}=0\Rightarrow$$
$$E\{[\mathbf{W}(\mathbf{y}-E\{\mathbf{y}\})-(\mathbf{f}-E\{\mathbf{f}\})](\mathbf{y}-E\{\mathbf{y}\})^{\mathbf{T}}\}=0$$

If $\tilde{\mathbf{y}}=\mathbf{y}-E\{\mathbf{y}\}$ and $\tilde{\mathbf{f}}=\mathbf{f}-E\{\mathbf{f}\}$ then

$$E\{(\mathbf{W}\tilde{\mathbf{y}}-\tilde{\mathbf{f}})\tilde{\mathbf{y}}^{\mathbf{T}}\}=0\Rightarrow$$
$$E\{\mathbf{W}\tilde{\mathbf{y}}\tilde{\mathbf{y}}^{\mathbf{T}}\}=E\{\tilde{\mathbf{f}}\tilde{\mathbf{y}}^{\mathbf{T}}\}\Rightarrow \mathbf{W}E\{\tilde{\mathbf{y}}\tilde{\mathbf{y}}^{\mathbf{T}}\}=E\{\tilde{\mathbf{f}}\tilde{\mathbf{y}}^{\mathbf{T}}\}\Rightarrow \mathbf{WR}_{\tilde{\mathbf{y}}\tilde{\mathbf{y}}}=\mathbf{R}_{\tilde{\mathbf{f}}\tilde{\mathbf{y}}}$$

If the original and the degraded image are both zero mean then $\mathbf{R}_{\tilde{\mathbf{y}}\tilde{\mathbf{y}}}=\mathbf{R}_{\mathbf{yy}}$ and $\mathbf{R}_{\tilde{\mathbf{f}}\tilde{\mathbf{y}}}=\mathbf{R}_{\mathbf{fy}}$. In that case we have that $\mathbf{WR}_{\mathbf{yy}}=\mathbf{R}_{\mathbf{fy}}$. If we go back to the degradation model and find the autocorrelation matrix of the degraded image then we get that

$$\mathbf{y}=\mathbf{Hf}+\mathbf{n}\Rightarrow \mathbf{y}^{\mathbf{T}}=\mathbf{f}^{\mathbf{T}}\mathbf{H}^{\mathbf{T}}+\mathbf{n}^{\mathbf{T}}$$
$$E\{\mathbf{yy}^{\mathbf{T}}\}=\mathbf{HR_{ff}H^T}+\mathbf{R_{nn}}=\mathbf{R_{yy}}$$

$$E\{\mathbf{f}\mathbf{y}^{\mathbf{T}}\} = \mathbf{R_{ff}}\mathbf{H}^{\mathbf{T}} = \mathbf{R_{fy}}$$

From the above we get the following results

$$\mathbf{W} = \mathbf{R_{fy}}\mathbf{R_{yy}}^{-1} = \mathbf{R_{ff}}\mathbf{H}^{\mathbf{T}}(\mathbf{H}\mathbf{R_{ff}}\mathbf{H}^{\mathbf{T}} + \mathbf{R_{nn}})^{-1}$$
$$\hat{\mathbf{f}} = \mathbf{R_{ff}}\mathbf{H}^{\mathbf{T}}(\mathbf{H}\mathbf{R_{ff}}\mathbf{H}^{\mathbf{T}} + \mathbf{R_{nn}})^{-1}\mathbf{y}$$

Note that knowledge of $\mathbf{R_{ff}}$ and $\mathbf{R_{nn}}$ is assumed. In frequency domain

$$W(u,v) = \frac{S_{ff}(u,v)H^*(u,v)}{S_{ff}(u,v)|H(u,v)|^2 + S_{nn}(u,v)}$$
$$\hat{F}(u,v) = \frac{S_{ff}(u,v)H^*(u,v)}{S_{ff}(u,v)|H(u,v)|^2 + S_{nn}(u,v)}Y(u,v)$$

Computational issues

The noise variance has to be known, otherwise it is estimated from a flat region of the observed image. In practical cases where a single copy of the degraded image is available, it is quite common to use $S_{yy}(u,v)$ as an estimate of $S_{ff}(u,v)$. **This is very often a poor estimate.**

Wiener smoothing filter

In the absence of any blur, $H(u,v) = 1$ and

$$W(u,v) = \frac{S_{ff}(u,v)}{S_{ff}(u,v) + S_{nn}(u,v)} = \frac{(SNR)}{(SNR)+1}$$

(i) $(SNR) >> 1 \Rightarrow W(u,v) \cong 1$

(ii) $(SNR) << 1 \Rightarrow W(u,v) \cong (SNR)$

(SNR) is high in low spatial frequencies and low in high spatial frequencies so $W(u,v)$ can be implemented with a lowpass (smoothing) filter.

5.1.3 Relation with inverse filtering

If $S_{nn}(u,v) = 0 \Rightarrow W(u,v) = \dfrac{1}{H(u,v)}$ which is the inverse filter

If $S_{nn}(u,v) \to 0 \Rightarrow \lim\limits_{S_{nn}\to 0} W(u,v) = \begin{cases} \dfrac{1}{H(u,v)} & H(u,v) \neq 0 \\ 0 & H(u,v) = 0 \end{cases}$

which is the pseudoinverse filter.

5.1.4 Iterative Wiener filters

They refer to a class of iterative procedures that successively use the Wiener filtered signal as an improved prototype to update the covariance estimates of the original image as follows.

Step 0: Initial estimate of $\mathbf{R_{ff}}$

$$\mathbf{R_{ff}}(0) = \mathbf{R_{yy}} = E\{\mathbf{yy^T}\}$$

Step 1: Construct the i^{th} restoration filter

$$\mathbf{W}(i+1) = \mathbf{R_{ff}}(i)\mathbf{H^T}(\mathbf{HR_{ff}}(i)\mathbf{H^T} + \mathbf{R_{nn}})^{-1}$$

Step 2: Obtain the $(i+1)^{\text{th}}$ estimate of the restored image

$$\hat{\mathbf{f}}(i+1) = \mathbf{W}(i+1)\mathbf{y}$$

Step 3: Use $\hat{\mathbf{f}}(i+1)$ to compute an improved estimate of $\mathbf{R_{ff}}$ given by

$$\mathbf{R_{ff}}(i+1) = E\{\hat{\mathbf{f}}(i+1)\hat{\mathbf{f}}^{\mathbf{T}}(i+1)\}$$

Step 4: Increase i and repeat steps 1,2,3,4.

<u>Recommended Questions</u>

1. Explain the importance process in image restoration process in image processing. Explain any four important noise probability density functions.
2. Discuss the importance of adaptive filters in image restoration system. Highlight the working of adaptive median filters.
3. Explain adaptive median filter and its advantages.
4. How do you reduce the periodic noise using frequency domain filters?
5. Derive the expression for observed image when the degradations are linear position invariant.
6. With a block diagram, briefly explain the image model of degradation-restoration process.
7. Explain notch reject filters. How can we obtain the notch filter that pass rather than suppressing the frequency in notch area?
8. Explain the Weiner filtering method of restoring images.

UNIT – 8

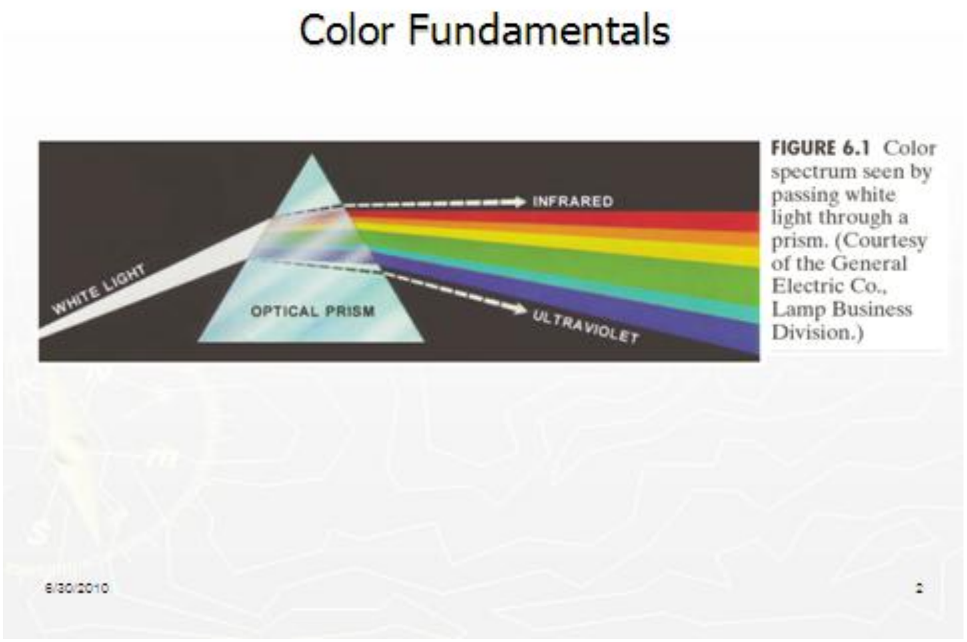
Color Fundamentals

The characteristics generally used to distinguish one color from another are brightness, hue, and saturation

brightness: the achromatic notion of intensity.

hue: dominant wavelength in a mixture of light waves, represents dominant color as perceived by an observer.

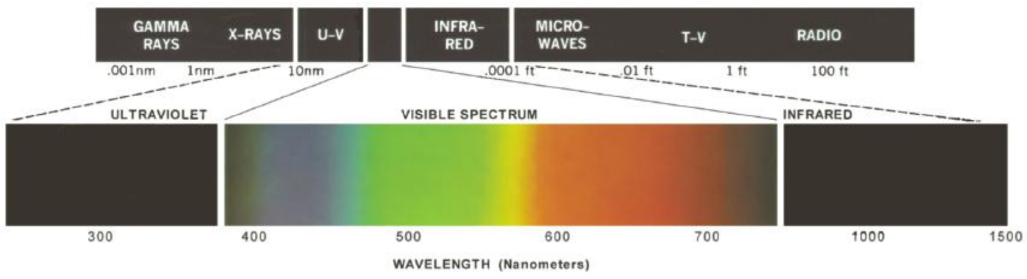
saturation: relative purity or the amount of white light mixed with its hue.

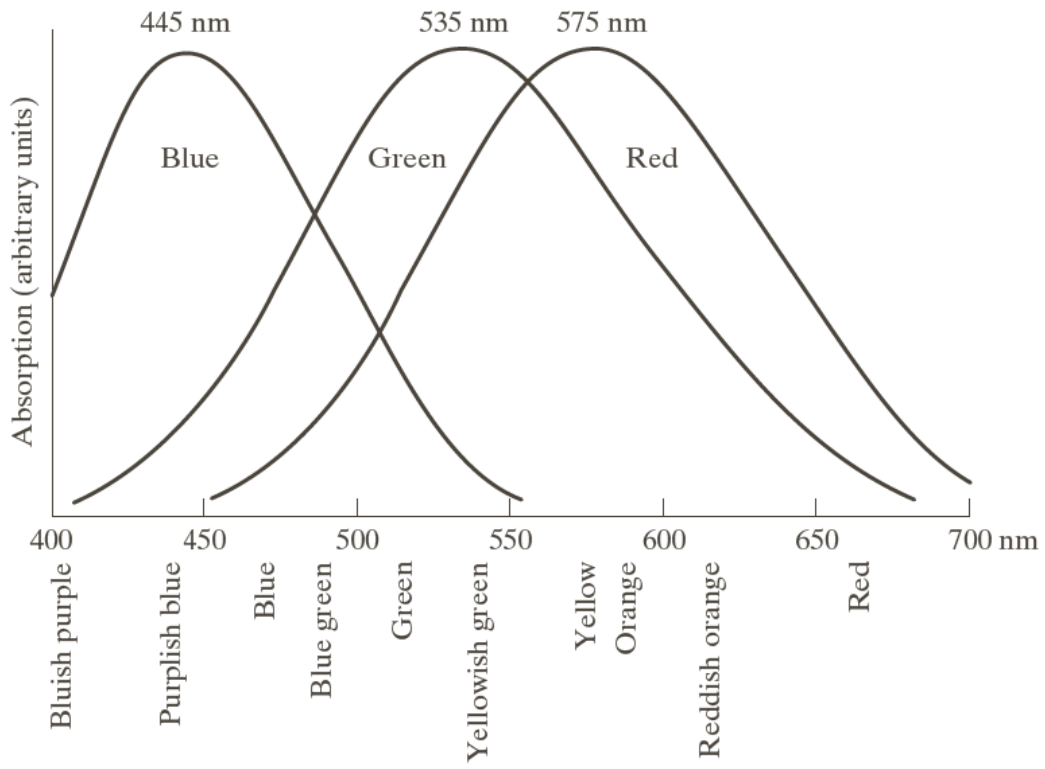


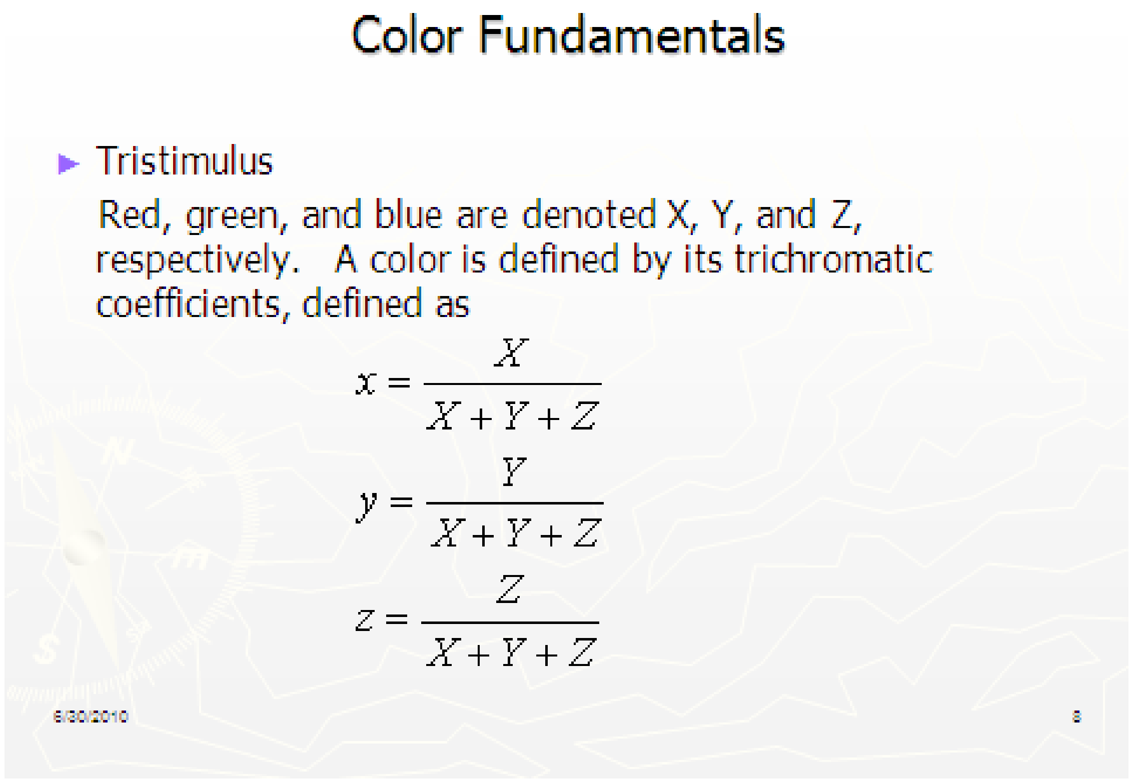
Color Fundamentals

FIGURE 6.1 Color spectrum seen by passing white light through a prism. (Courtesy of the General Electric Co., Lamp Business Division.)

8/30/2010 2







Color Fundamentals

- Tristimulus

  Red, green, and blue are denoted X, Y, and Z, respectively. A color is defined by its trichromatic coefficients, defined as

$$x = \frac{X}{X+Y+Z}$$

$$y = \frac{Y}{X+Y+Z}$$

$$z = \frac{Z}{X+Y+Z}$$

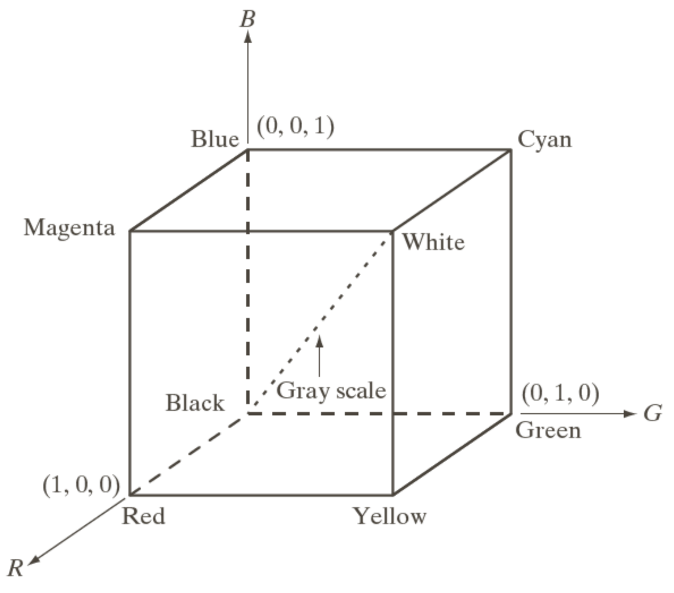
6/30/2010 8

Color Models

The purpose of a color model (also called *color space* or *color system*) is to facilitate the specification of colors in some standard, generally accepted way. In essence, a color model is a specification of a coordinate system and a subspace within that system where each color is represented by a single point.

Most color models in use today are oriented either toward hardware (such as for color monitors and printers) or toward applications where color manipulation is a goal (such as in the creation of color graphics for animation). In terms of digital image processing, the hardware-oriented models most commonly used in practice are the RGB (red, green, blue) model for color monitors and a broad class of color video cameras; the CMY (cyan, magenta, yellow) and CMYK (cyan, magenta, yellow, black) models for color printing; and the HSI (hue, saturation, intensity) model, which corresponds closely with the way humans describe and interpret color. The HSI model also has the advantage that it decouples the color and gray-scale information in an image, making it suitable for many of the gray-scale techniques developed in this book. There are numerous color models in use today due to the fact that color science is a broad field that encompasses many areas of application. It is tempting to dwell on some of these models here simply because they are interesting and informative. However, keeping to the task at hand, the models discussed in this chapter are leading models for image processing. Having mastered the material in this chapter, the reader will have no difficulty in understanding additional color models in use today.

The RGB Color Model



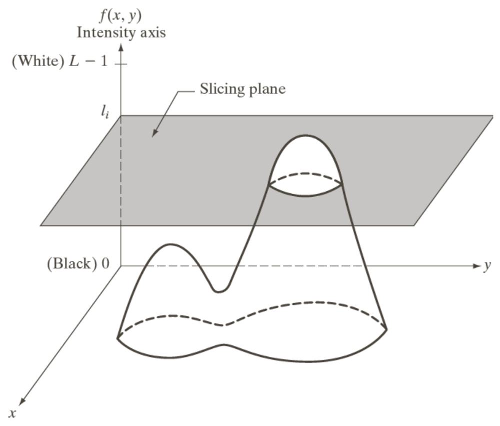
In the RGB model, each color appears in its primary spectral components of red, green, and blue. This model is based on a Cartesian coordinate system. The color subspace of interest is the cube shown in Fig. 6.7, in which RGB values are at three corners; cyan, magenta, and yellow are at three other corners; black is at the origin; and white is at the corner farthest from the origin. In this model, the gray scale (points of equal RGB values) extends from black to white along the line joining these two points. The different colors in this model are points on or inside the cube, and are defined by vectors extending from the origin. For

Images represented in the RGB color model consist of three component images, one for each primary color. When fed into an RGB monitor, these three images combine on the phosphor screen to produce a composite color image. The number of bits used to represent each pixel in RGB space is called the *pixel depth*. Consider an RGB image in which each of the red, green, and blue images is an 8-bit image. Under these conditions each RGB *color* pixel [that is, a triplet of values (R, G, B)] is said to have a depth of 24 bits (3 image planes times the number of bits per plane). The term *full-color* image is used often to denote a 24-bit RGB color image. The total number of colors in a 24-bit RGB image is $(2^8)^3 = 16{,}777{,}216$. Figure 6.8 shows the 24-bit RGB color cube corresponding to the diagram in Fig. 6.7.

Intensity Slicing

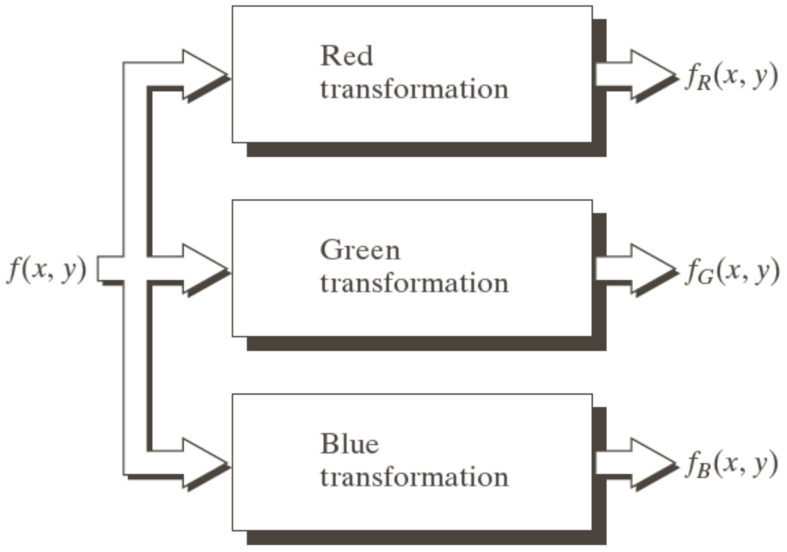
The technique of *intensity* (sometimes called *density*) *slicing* and color coding is one of the simplest examples of pseudocolor image processing. If an image is interpreted as a 3-D function (intensity versus spatial coordinates), the method can be viewed as one of placing planes parallel to the coordinate plane of the image; each plane then "slices" the function in the area of intersection. Figure 6.18 shows an example of using a plane at $f(x, y) = l_i$ to slice the image function into two levels.

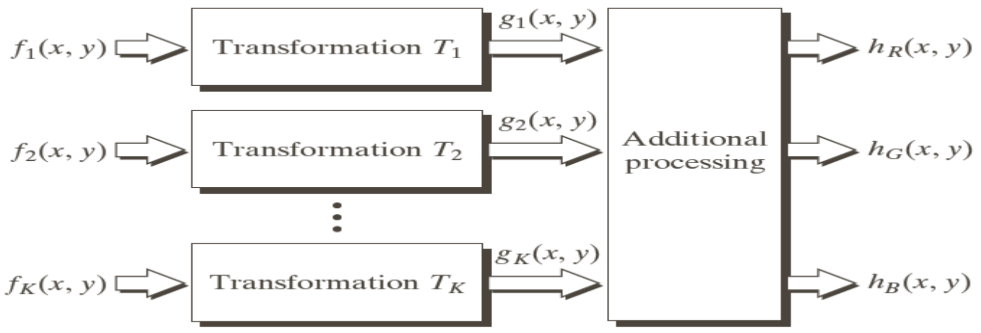
If a different color is assigned to each side of the plane shown in Fig. 6.18, any pixel whose gray level is above the plane will be coded with one color, and any pixel below the plane will be coded with the other. Levels that lie on the plane



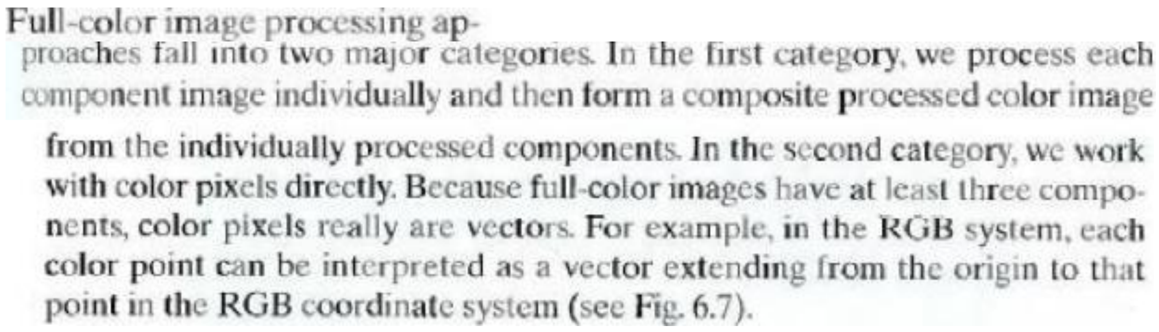
Pseudo color Image Processing.,

Gray level to Color Transformation





processing basics of full color image processing

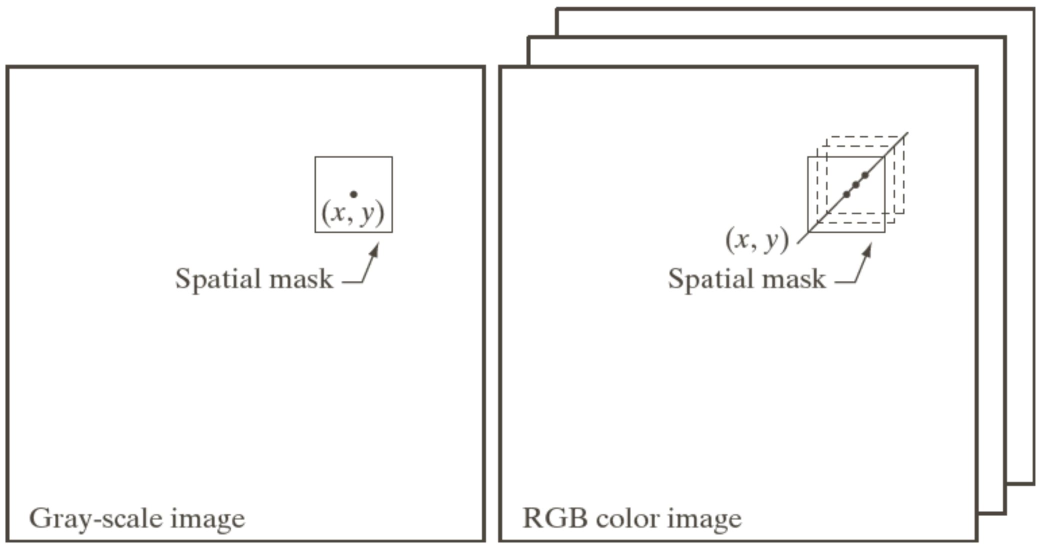

Full-color image processing approaches fall into two major categories. In the first category, we process each component image individually and then form a composite processed color image from the individually processed components. In the second category, we work with color pixels directly. Because full-color images have at least three components, color pixels really are vectors. For example, in the RGB system, each color point can be interpreted as a vector extending from the origin to that point in the RGB coordinate system (see Fig. 6.7).

Let c represent an arbitrary vector in RGB color space:

$$c = \begin{bmatrix} c_R \\ c_G \\ c_B \end{bmatrix} = \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

At coordinates (x, y),

$$c(x, y) = \begin{bmatrix} c_R(x, y) \\ c_G(x, y) \\ c_B(x, y) \end{bmatrix} = \begin{bmatrix} R(x, y) \\ G(x, y) \\ B(x, y) \end{bmatrix}$$

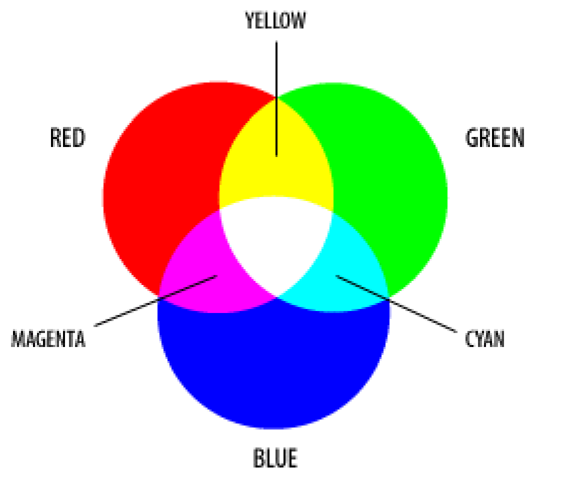


RGB

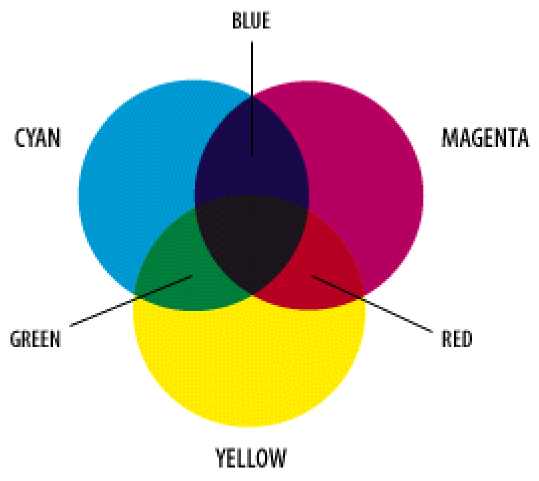
The RGB colour model relates very closely to the way we perceive colour with the r, g and b receptors in our retinas. RGB uses additive colour mixing and is the basic colour model used in television or any other medium that projects colour with light. It is the basic colour model used in computers and for web graphics, but it cannot be used for print production.

The secondary colours of RGB – cyan, magenta, and yellow – are formed by mixing two of the primary colours (red, green or blue) and excluding the third colour. Red and green combine to make yellow, green and blue to make cyan, and blue and red form magenta. The combination of red, green, and blue in full intensity makes white.

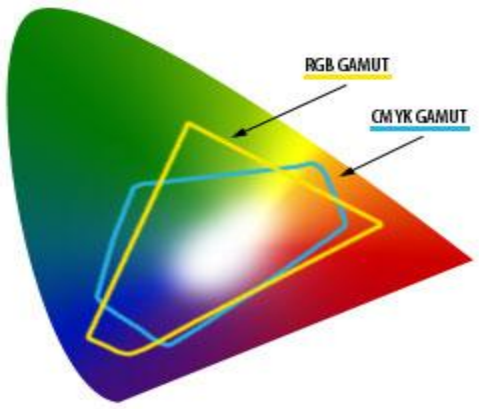
In Photoshop using the “screen” mode for the different layers in an image will make the intensities mix together according to the additive colour mixing model. This is analogous to stacking slide images on top of each other and shining light through them.



CMYK The 4-colour CMYK model used in printing lays down overlapping layers of varying percentages of transparent cyan (C), magenta (M) and yellow (Y) inks. In addition a layer of black (K) ink can be added. The CMYK model uses the subtractive colour model.



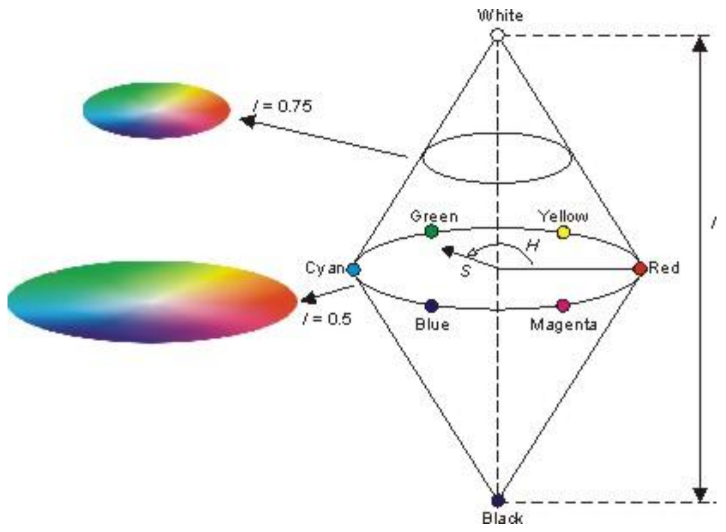
Gamut The range, or gamut, of human colour perception is quite large. The two colour spaces discussed here span only a fraction of the colours we can see. Furthermore the two spaces do not have the same gamut, meaning that converting from one colour space to the other may cause problems for colours in the outer regions of the gamuts.



The HSI color space

The HSI color space is very important and attractive color model for image processing applications because it represents color s similarly how the human eye senses colors.

The HSI color model represents every color with three components: hue (H), saturation (S), intensity (I). The below figure illustrates how the HIS color space represents colors.



The Hue component describes the color itself in the form of an angle between [0,360] degrees. 0 degree mean red, 120 means green 240 means blue. 60 degrees is yellow, 300 degrees is magenta.

The Saturation component signals how much the color is polluted with white color. The range of the S component is [0,1].

The Intensity range is between [0,1] and 0 means black, 1 means white.

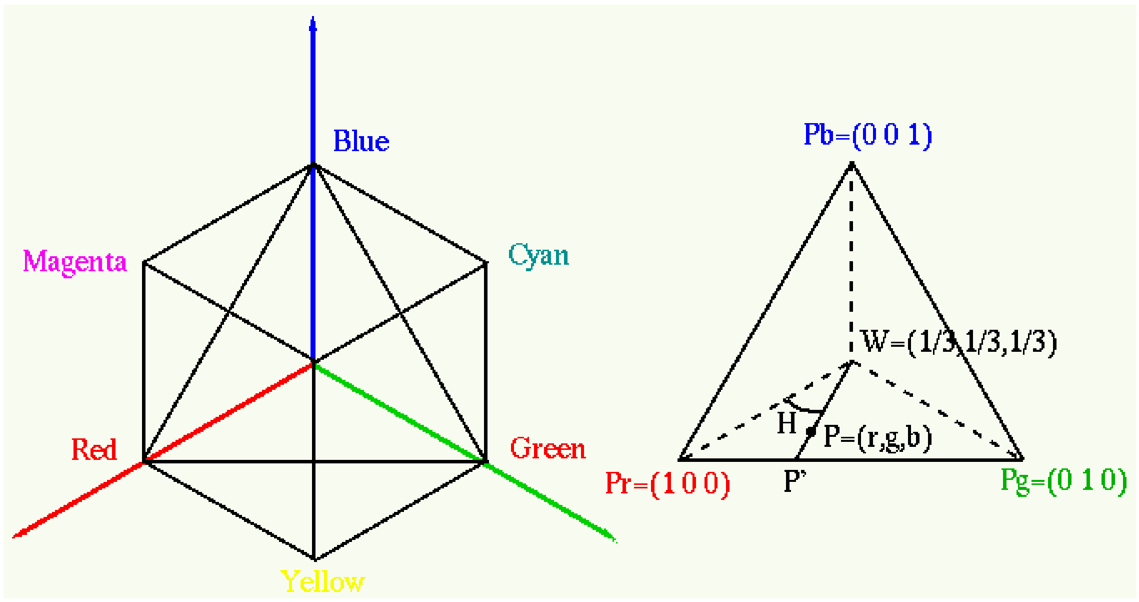
As the above figure shows, hue is more meaningful when saturation approaches 1 and less meaningful when saturation approaches 0 or when intensity approaches 0 or 1. Intensity also limits the saturation values.

To formula that converts from RGB to HSI or back is more complicated than with other color models, therefore we will not elaborate on the detailed specifics involved in this process.

RGB	CMY	CMYK
HSI	HSV	L*a*b
XYZ	YIQ	YUV

Conversion from RGB to HSI

Given the intensities of the three primaries RGB of a color, we can find its HSV representation using different models. Here we use the RGB plane of the cube to find the corresponding HSV. The three vertices are represented by P_r, P_g and P_b, and the three components of the given color is represented by a 3D point $P=(R,G,B)$. We also assume the intensities are normalized so that the R, G and B values are between 0 and 1, so that point P is inside or on the surface of the color cube.



- Determine the intensity I:

One of the definitions of intensity is

$$I \triangleq \frac{1}{3}(R+G+B)$$

- Determine the hue H:

First find the intersection of the color vector (R,G,B) with the RGB triangle $R+G+B=1$:

$$\begin{cases} r \triangleq R/(R+G+B) = R/3I \\ g \triangleq G/(R+G+B) = G/3I \\ b \triangleq B/(R+G+B) = B/3I \end{cases}$$

This point $p=(r,g,b)$ is on the RGB triangle as $r+g+b=1$. Here we assume the point p is inside the triangle formed by points w, P_r, and P_g. The hue is the angle $\angle H$ formed by the vectors $\overline{pw}$ and $\overline{P_r w}$. Consider the dot product of these two vectors:

$$\overline{P_r w} \cdot \overline{pw} = (P_r - w) \cdot (p - w) = |P_r - w|\,|p - w|\; cos\angle H$$

where $P_r = (1,0,0)$, $p = (r,g,b)$ and $w = (1/3, 1/3, 1/3)$, and

$$(P_r - w) = (1,0,0) - (1/3,1/3,1/3) = (2/3, -1/3, -1/3)$$

$$(p - w) = (r - 1/3, g - 1/3, b - 1/3)$$

$$(P_r - w)\cdot(p - w) = \frac{2}{3}(r - \frac{1}{3}) - \frac{1}{3}(g - \frac{1}{3}) - \frac{1}{3}(b - \frac{1}{3}) = 2r - g - b = \frac{2R - G - B}{3(R+G+B)}$$

$$|P_r - w| = \sqrt{(1 - \frac{1}{3})^2 + (0 - \frac{1}{3})^2 + (0 - \frac{1}{3})^2} = \sqrt{\frac{2}{3}}$$

$$|p - w| : \sqrt{(r - \frac{1}{3})^2 + (g - \frac{1}{3})^2 + (b - \frac{1}{3})^2} = \sqrt{\frac{9(R^2 + G^2 + B^2) - 3(R + G + B)^2}{9(R + G + B)^2}}$$

$$: \frac{\sqrt{6(R^2 + G^2 + B^2 - RG - GB - BR)}}{3(R + G + B)}$$

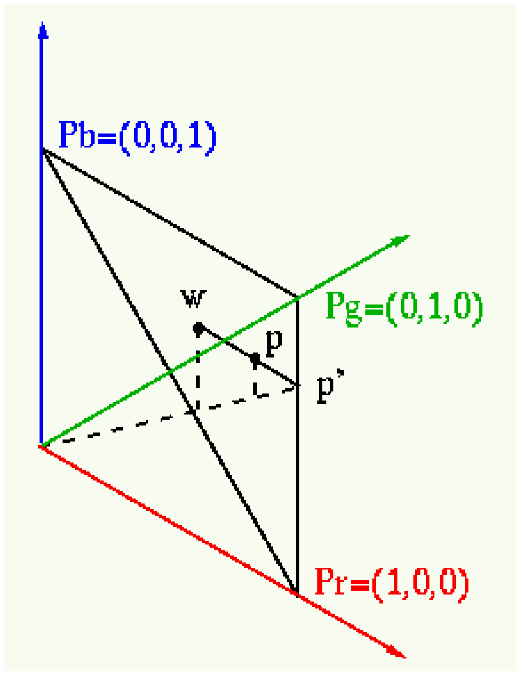
Now the hue angle can be found to be

$$\angle H : cos^{-1}[\frac{(P_r - w)\cdot(p - w)}{|P_r - w|\,|p - w|}] = cos^{-1}[\frac{2R - G - B}{\sqrt{2/3}\sqrt{6(R^2 + G^2 + B^2 - RG - GB - BR)}}]$$

$$: cos^{-1}[\frac{3R - (R + G + B)}{2\sqrt{R^2 + G^2 + B^2 - RG - GB - BR}}] = cos^{-1}[\frac{(R - G) + (R - B)}{2\sqrt{(R - G)^2 + (R - B)(G - B)}}]$$

If $B > G$, then $\angle H = 360 - \angle H$.

- Determine S:



The saturation of the colors on any of the three edges of the RGB triangle is defined as 1 (100% saturated), and the saturation of $w=(1/3,1/3,1/3)$ is zero. Denote p' as the intersection of the extension of line $\overline{wp}$ with the edge. If the normalized color is $p=w$, $S_p=0$, and if $p=p'$, $S_p=1$. The saturation of any color point p between w and p' is defined as

$$S_p = \frac{|wp|}{|wp'|} = \frac{|wp'|-|pp'|}{|wp'|} = 1-\frac{|pp'|}{|wp'|} = 1-\frac{b}{1/3} = 1-3b$$

Here it is assumed that point p is inside the triangle $\triangle P_r w P_g$ so that $b = min(r,g,b)$.

In general

$$S_p = 1-3\,min(r,g,b) = \begin{cases} 0, & min(r,g,b) = 1/3, \text{ i.e. } r = g = b = 1/3, \text{ i.e. } p = w \\ 1, & min(r,g,b) = 0, \text{ i.e. } p \text{ is on one of the edges} \\ 0 < S_p < 1, & 0 < min(r,g,b) < 1/3 \end{cases}$$

Or

Conversion from HSI to RGB

Consider three possible cases in terms of the hue angle $\angle H$:

- $0 < \angle H < 120$ (p inside $\triangle P_r w P_g$)

From $S = 1 - 3b$, we get

$$b = (1 - S)/3$$

Also we can get

$$r = \frac{1}{3}[1 + \frac{S cos \angle H}{cos(60 - \angle H)}]$$

and

$$g = 1 - b - r$$

Given r, g, b, we can get R, G, B from I. As

$$r = \frac{R}{R + G + B} = \frac{R}{3I}$$

we have

$$R = 3Ir, \quad G = 3Ig, \quad B = 3Ib$$

- $120 < \angle H < 240$ (p inside $\triangle P_g w P_b$)

$$\angle H = \angle H - 120$$

$$r = (1 - S)/3$$

$$g = \frac{1}{3}[1 + \frac{S cos \angle H}{cos(60 - \angle H)}]$$

$$b = 1 - g - r$$

- $240 < \angle H < 360$ (p inside $\triangle P_b w P_r$)

$$\angle H = \angle H - 240$$

$$g = (1 - S)/3$$

$$b = \frac{1}{3}[1 + \frac{S cos \angle H}{cos(60 - \angle H)}]$$

$$r = 1 - g - b$$

CMYK_CONVERT

The CMYK_CONVERT procedure converts from the CMYK (cyan-magenta-yellow-black) color model to RGB (red-green-blue) and vice versa.

The procedure uses the following method to convert from CMYK to RGB:

R = (255 - C) (1 - K/255)

G = (255 - M) (1 - K/255)

B = (255 - Y) (1 - K/255)

To convert from RGB to CMYK, the procedure uses the following method:

K = minimum of (R, G, B)

C = 255 [1 - R/(255 - K)] (if K=255 then C=0)

M = 255 [1 - G/(255 - K)] (if K=255 then M=0)

Y = 255 [1 - B/(255 - K)] (if K=255 then Y=0)

In both cases the CMYK and RGB values are assumed to be in the range 0 to 255.

Note
There is no single method that is used for CMYK/RGB conversion. The method used by CMYK_CONVERT is the simplest and, depending on printing inks and screen colors, might not be optimal in all situations.

This routine is written in the IDL language. Its source code can be found in the file cmyk_convert.pro in the libsubdirectory of the IDL distribution.

Syntax

CMYK_CONVERT, C, M, Y, K, R, G, B [, /TO_CMYK]

Arguments

C, M, Y, K

To convert from CMYK to RGB, set these arguments to scalars or arrays containing the CMYK values in the range 0-255. To convert from RGB to CMYK (with the TO_CMYK keyword set), set these arguments to named variables that will contain the converted values.

R, G, B

To convert from CMYK to RGB, set these arguments to named variables that will contain the converted values. To convert from RGB to CMYK (with the TO_CMYK keyword set), set these arguments to scalars or arrays containing the RGB values.

Keywords

TO_CMYK

If this keyword is set, the values contained in the RGB arguments are converted to CMYK. The default is to convert from CMYK to RGB.

<u>**Recommended Questions**</u>

1. Explain the colour models.
2. Explain the following order statistics filters, indicating their uses.

i) Median filter ii) max filter iii) min filter.

3. Explain the RGB colour model.
4. Write a note on the following pseudo image processing techniques.

i) Intensity slicing

ii) Graylevel to colour transformations.

5. Write steps involved in converting colours from RGB to HSI and vice versa.
6. Explain pseudocolour image processing in brief.
7. Write short notes on i) weiner filtering ii) Inverse filtering